UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering,
Mathematics & Computer Science

Automatic Short Answer Grading using Text-to-Text Transfer Transformer Model

Stefan Haller

M.Sc. Thesis in Business Information Technology
Specialization Data Science
October 2020

Supervisors:
Dr. Christin Seifert
Dr. Nicola Strisciuglio
Dr. Adina Aldea

Telecommunication Engineering Group
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Acknowledgment

This thesis is the result of the last two years. I am grateful for every single step I took to get to this point. I am grateful and happy for the time I spent on learning completely new things which I never thought I would learn. It was one of the most intense and amazing times in my life and for my personal development. I would like to express my gratitude to the people who supported me over the years.

This work represents for me an end and a new beginning and it has been an intense journey I have gone through. I was enthusiastic about research and thus dived myself deeply into the topics NLP, Deep Learning and ASAG. My personal interest in these topics has been further increased and I am very grateful for that. Many difficulties and challenges on the way were eased by the support of my supervisors. Therefore I would like to thank Christin Seifert, Nicola Strisciuglio and Adina Aldea for their interest, support and guidance with my work.

I would like to thank you all for your advice and ideas. Especially for the honest and helpful feedback on how to not lose focus and on how to conduct research. Christin, I want to thank you especially for your ideas and suggestions on how I can improve my work. Nicola, I would like to thank you especially for your tips regarding the approach and the experiments. And last but not least I want to thank you Adina for taking the time to validate my data and helping me enormously with the data collection.

Despite the guidance and feedback, you always gave me the freedom to bring all my own ideas and visions into the work. Therefore, I appreciate your approach as supervisors where you steered me in the right direction whenever you thought I needed it.

Finally, I want to express my gratitude to my parents and to my friends, especially my friends from the University of Twente for providing me with support and encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without you. Thank you.

Abstract

We explore in this study the effects of multi-task training methods and domain adaptation on Automatic Short Answer Grading (ASAG) using the text-to-text transfer transformer model (T5). Within this study, we design an ASAG model and evaluate its applicability to a practice dataset from the University of Twente. We fine-tuned a multi-task model that is trained on a profound selection of related tasks and an extensively pre-trained model. We evaluate the performance of the models on the SciEntsBank dataset and achieved new-state-of-the-art results. With the best performing model we showed that domain-independent fine-tuning is preferable to domain-specific fine-tuning for data sparse cases. The optimized model was used and its performance was demonstrated in the university context. The predictions of the model were explained with different model-agnostic methods which resulted in several hypotheses that describe the model behavior. The reported results reveal that the model is biased towards correct answers and has particular problems with partially correct answers. Through the gained knowledge about the decision behavior, the model robustness against student manipulations was evaluated and tested. Within a validation study, we asked students to generate manipulation answers. Our findings emphasize the susceptibility of the model towards manipulation strategies and difficulties with handling imbalanced and sparse data. We observe that for a functional ASAG model balanced and extensive data are necessary.

Abbreviations and Acronyms

ASAG	Automatic Short Answer Grading
ASAP-SAS	Automated Student Assessment Prize Short Answer Scoring
AI	Artificial Intelligence
BERT	Bidirectional Encoder Representations using Transformers
BiLSTM	Bidirectional Long Short-Term Memory
C4	Colossal Clean Crawled Corpus
CNN	Convolutional Neural Network
GLUE	General Language Understanding Evaluation
L2X	Learning to Explain
LSTM	Long Short-Term Memory
ML	Machine Learning
MTL	Multi-task Learning
NER	Named Entity Recognition
NLI	Natural Language Inference
NLP	Natural Language Processing
POS	Part-of-Speech
RAKE	Rapid Automatic Keyword Extraction
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Networks
SQuAD	Stanford Question Answering Dataset
T5	Text-to-Text Transfer Transformer

List of Figures

List of Tables

Contents

Chapter 1

Introduction

Now more than ever, the question is being asked whether the current school system has adapted to fit future demands. Although technological advances have drastically changed most areas of life, the educational system has not progressed in proportion. In essence, there is still one individual facilitating the classroom environment, whether in-person or online. In current years, student's desire a wider range of information available to them. However, each student learns at a different pace and individuality is not supported properly by the current educational system. The use of tutors and educational content (e.g. Khan Academy) has therefore increased steadily over the years to meet this demand of learning not provided in the classroom. The Internet was used to pioneer the creation of such digital developments in education.

But what does the future of education look like and what will be the next step to approach asynchronous learning?

Let's imagine the perfect school for the future generations where everybody has the same educational possibilities with its own tutor that aligns the learning pace according to the capabilities of the student. Such a system raises several questions and from today's perspective is connected with various problems. On the one hand, there are not enough people to teach each child separately and this would be far too expensive under the current conditions. On the other hand, the quality and the individuality of the tutors are different which leads to inequalities and non-individual education. The first problem has been largely solved by technological advancements. Today, everyone can access knowledge and further education from anywhere. People even have the possibility to study online courses of renowned universities for almost free. As a consequence of these developments, more people evolved into digital teachers by teaching their knowledge online. However, this resulted in an almost abundance of information and content which leaves the internet as a library of many educational videos and content. Regardless of the benefits, it did not solve the problem of the individuality of education. It is obvious that such a problem might be solved with the coming revolution in the education sector with an individual digital tutor. Such a tutor will embody a diversity of skills and characteristics but it is not set in stone how they will look like. However, the assumption is obvious that a digital tutor might provide student's with customized content (e.g. suggesting suitable learning videos) while simultaneously tests them on what they know. More importantly, it will adapt to the way the student learns over time by comparing the effectiveness of different videos and different tests to decide what works best for the student. This allows student's to be individually supervised and their learning process to be tailored to their abilities with very little human interference.

In the past years, many approaches have been developed that point in this direction. Each of the approaches relies on the development in the field of artificial intelligence (AI) since it is crucial for developing such a system. Therefore, AI will play the most significant role and pave the way for the next revolution in the education system.

Consequently, the question arises on how much progress has been made and what steps can be taken now to further develop and get closer to future a digital tutor.

1.1 Motivation and Problem Statement

To address this question the entire prospective tutoring system must be broken down into the individual parts and approached chronologically. For such a system it is decisive to analyze the learning process of the student to provide the student with individual learning advice. This makes the evaluation of the learning success of the student crucial and perspectively important to solve.

The evaluation of student's learning process is already one of the most critical points in the school system since it describes the efficiency and success of acquiring knowledge. For such an evaluation process it is crucial to assess the learned knowledge of the student as precisely as possible. Currently, the major assessment methods used are exams [4]. In these exams, the knowledge of the student's can be tested in different ways. Testing methods vary in general and range from closed answers (e.g. multiple-choice) to open answers (e.g. essays or short answers) [5], each with advantages over the other. When it comes to a qualitative assessment of student knowledge, multiple-choice questions is not the most suitable assessment method since it only produces quantitative data and no qualitative. In contrast, open questions force the student to provide a compact description of his knowledge. This makes such questions the preferable choice since it captures the gained knowledge more precisely.

From research and technical perspective, this leads us into the area known as Automatic Short Answer Grading (ASAG). In this field, we define short answers according to [5]. According to their definition, an answer must fulfill five criteria to be considered as a short answer.

1. The question asks for external knowledge which means that the student is expected to answer by using his knowledge and not just passages from a provided prompt text
2. The student response needs to be given in a natural language
3. The length of the answer is around 50 words but not more than 100
4. The grading of the answer focuses on the content rather than the writing quality

5. The question restricts the student in his possible answers

In this field, natural language answers are evaluated on an ordinal scale which reflects the nature of a digital tutor system. But where are we in this field, how far is the development and where is it lacking?

A closer look at the literature on ASAG shows that there are hardly any holistic approaches where a realistic application is a final goal. This highlights that the research is lagging behind in topics that are essential for such an application. In detail, when taking a step back and looking at the whole context of the problem, topics like model interpretability and explainability are rarely addressed in the literature. Despite their importance for ASAG, hardly anyone takes the trouble to analyze the developed models. However, researchers did realize that a real model implementation requires the knowledge of the underlying decision basis. Otherwise, one is confronted with accountability problems which prevent an implementation. Another point is that most researchers approach the task only selectively by aiming for a good performance on some dataset. Most authors are satisfied with good test results for a given dataset. Hardly anyone is going one step further and is considering a practical demonstration and evaluation of such a model on different datasets. Consequentially, models lack in general applicability which makes the practical application improbable.

These shortcomings are further connected with the fact that researchers exclusively evaluate performances on respective test scores and thus fail to deal with the data sets and its characteristics. One reason for this is that ASAG itself is a data sparse field and therefore the data basis is not very extensive and structured. This sparsity also prevents progress in the field of adversarial attacks and makes models particularly susceptible to targeted manipulation attempts.

On the basis of this current status and the resulting shortcomings, requirements for such an ASAG model can be derived, which have to be solved to enable a practical implementation and advancement in the development of a digital tutor system (table 1.1).

Table 1.1: Identified general requirements for a ASAG model

Nr.	Requirement
1	The model is required to have high-performance and efficiency
2	The model needs to be trainable with small data while keeping performance
3	The model predictions are required to be comprehensible and explainable
4	The model needs to be robust enough to deal with student manipulations

These requirements raise the question of the extent to which currently available approaches can be used to meet them and what are next steps to further advance the development. Answering and evaluating these general questions is essential for advancements for digital tutors and are therefore the main motivation for this thesis.

1.2 Research Goal

Inspired by the latest developments around the field of ASAG the main goal of this thesis is to design an ASAG model and evaluate its applicability to practice. By demonstrating and evaluating such an implementation we further aim to gain valuable insights and to reveal the potential for improvements.

In order to achieve this, we formulated different objectives for each of the identified requirements. As table 1.2 shows, a model must be created that can be trained efficiently while keeping high-performance in short answer grading. Furthermore, the model is required to be able to compensate for the data sparse nature of the ASAG field. In addition, we want to make the model explainable and analyze the prediction behavior. To improve the practicability of the model, we demonstrate the model on a dataset from the University of Twente. The final goal of this work is to investigate how robust the model is.

Table 1.2: Derived research goals from requirements

Nr.	Research Goals
1	Create a model with an efficient training approach and high-performance
2	Create a model that deals efficiently with data sparsity
3	Make the model decisions comprehensible and explainable
4	Evaluate the model robustness on handling student manipulations

1.3 Research Questions

Within the scope of this work, research questions were defined that will be answered by the methodology and the particularly defined experiments. The research questions are aligned with the formulated objectives and thereby contribute to the achievement of the main goal. In detail, we can break the goals down into four relevant areas. These were addressed in the 4 main research questions and corresponding sub-questions:

1. Research Question: Does multi-task learning improve the performance of Automatic Short Answer Grading?

This question can be answered with two sub-questions.

1. Sub-question 1: Is a multi-task learning approach beneficial when incorporating datasets from the same and related research fields?

   To develop such an ASAG model it is important that the training process is efficient while aiming for the best possible performance. With this question, we analyze if the multi-task training approach is beneficial for the problem and if it can be further optimized by using a more profound dataset selecting process. This may reveal the potential for improving the training process by select specific datasets for multi-task pre-training.

2. Sub-question 2: Does a mulit-task pre-trained model improve Automatic Short Answer Grading and outperform the baseline?

   The next step is to increase the performance further by fine-tuning a more comprehensive model. A comparison with the previous model provides whether the pre-trained model or the self-trained model is more suitable for the ASAG task. This result can be used to determine the preferable model training approach.

2. Research Question: Does domain-specific fine-tuning influence the performance of Automatic Short Answer Grading?

This question aims to further optimize the fine-tuning process of the selected model by means of domain adaptation. It is essential to determine if domain-specific fine-tuning is beneficial or if it makes sense to include other unspecific data from different domains. This information is useful when deciding between either fine-tuning one model on several questions on multiple-source domains, or fine-tuning domain-specific models. Such a comparison reveal insights on how the model can be optimized with sparse data. This results in a preferred fine-tuning process and can be used as a basis for the demonstration and evaluation.

3. Research Question: How can we explain model decisions in a real-world application?

The knowledge gained from the first two research questions regarding model training and the fine-tuning process can be applied to a real-world dataset from the University of Twente. After successful testing and evaluation, the question focuses on making the model decisions comprehensible and explainable by applying useful algorithms. Based on the findings in the literature review an integrated method compilation is introduced to explain the model behavior with hypotheses.

4. Research Question: How robust is the model towards student manipulations?

This question follows up on the hypotheses found in the previous question by using them to analyze and evaluate the model's robustness against student manipulations. The model robustness is evaluated by generating adversarial answers which challenge the model. This results in an assessment of the extent to which the model is susceptible to student manipulations which provides insights about the deployment possibility in a real-world setting like the university.

1.4 Research Methodology

The following research methodology serves as a guideline for the thesis and describes the overall research structure, the respective roles of the research questions, and their interrelationships with the overall goal of creating a real-world ASAG model. The detailed implementation of the mentioned points are described in detail in the methodology (see chapter 4).

For our research we used the illustrated process in figure 1.1. It encompasses several main activities: problem identification and motivation, identification of specific requirements to derive specific model objectives, design and optimization of the training and fine-tuning method as well as model demonstration and evaluation.

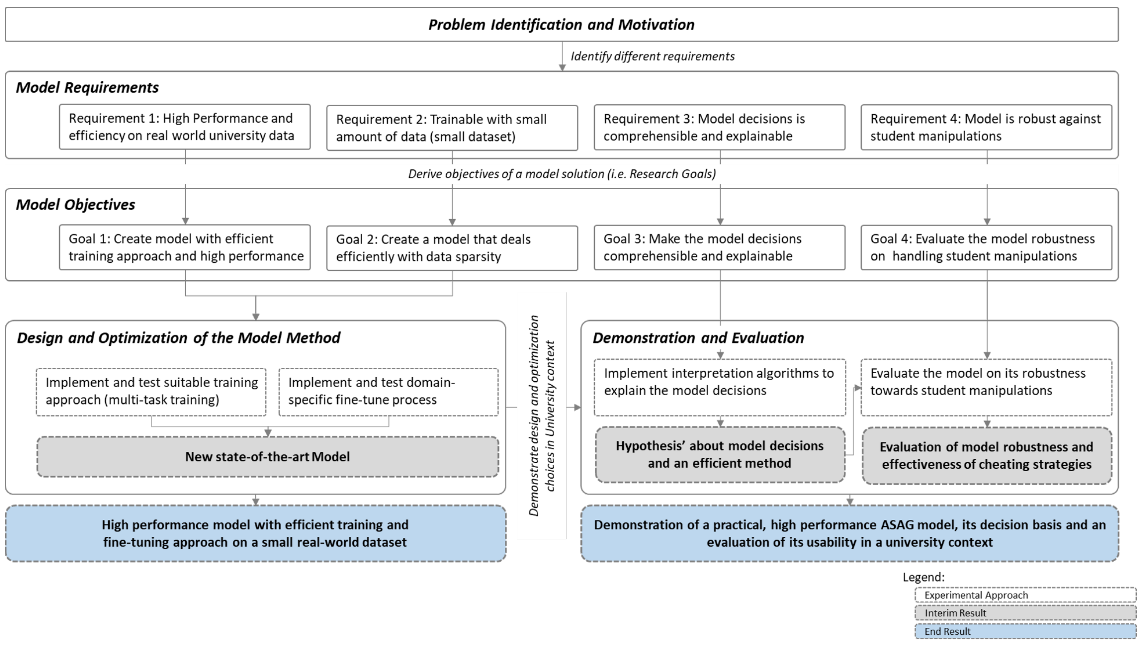

Problem Identification and Motivation
Identify different requirements
Model Requirements
Requirement 1: High Performance and efficiency on real world university data
Requirement 2: Trainable with small amount of data (small dataset)
Requirement 3: Model decisions is comprehensible and explainable
Requirement 4: Model is robust against student manipulations
Derive objectives of a model solution (i.e. Research Goals)
Model Objectives
Goal 1: Create model with efficient training approach and high performance
Goal 2: Create a model that deals efficiently with data sparsity
Goal 3: Make the model decisions comprehensible and explainable
Goal 4: Evaluate the model robustness on handling student manipulations
Design and Optimization of the Model Method
Implement and test suitable training approach (multi-task training)
Implement and test domain-specific fine-tune process
New state-of-the-art Model
High performance model with efficient training and fine-tuning approach on a small real-world dataset
Demonstrate design and optimization choices in University context
Demonstration and Evaluation
Implement interpretation algorithms to explain the model decisions
Evaluate the model on its robustness towards student manipulations
Hypothesis' about model decisions and an efficient method
Evaluation of model robustness and effectiveness of cheating strategies
Demonstration of a practical, high performance ASAG model, its decision basis and an evaluation of its usability in a university context
Legend:
Experimental Approach
Interim Result
End Result

Figure 1.1: Overview of the research methodology

With the chosen research approach we represent the main research goal of designing a real-world ASAG model and evaluating its applicability in practice. For designing such a model we first illustrate the problem context and motivation (section 1.1. From this, specific model requirements for a real-world ASAG model are identified. Based on these requirements specific objectives are derived which are the foundation of our model design. We identify suitable models, training and evaluation methods, and algorithms for each of the objectives by conducting a semi-structured literature review in chapter 3.

To address the different characteristics and the actual model design the work is structured in four pillars. Each of the pillars represents one requirement and a corresponding goal. The first two pillars are used to identify and determine the preferable model training method and fine-tune process by means of a public ASAG dataset which is described in section 5.1. In order to design a model that contributes to achieving the goal of high-performance and efficiency on small data, we analyzed first to what extend multi-task training can be optimized and if we can design a new state-of-the-art model. This was done by answering the first research question with the two corresponding sub-questions which resulted in a high-performance model with a preferred training method.

As a next step, this model was used to investigate if the fine-tuning process can be improved by means of domain adaptation. This answers the question to what extent a data sparse, domain-specific fine-tune process or a non-domain-specific fine-tune process is superior with respect to the performance (research question 2). After identifying the best suitable model and fine-tune process the performance on a real-world dataset from the University of Twente has been demonstrated and evaluated. This led to the third pillar where the model decisions have been made comprehensible and explainable. The aforementioned pillar has been answered with the third research question which introduced a composite approach of different methods to make the model explainable. As a result, different hypotheses that explain the model decision have been constructed.

Through the gained knowledge about the decision behavior, the model robustness against student manipulations was then evaluated and tested within research question four. This was achieved by defining individual adversarial attacks from a experimental group of students and attacks that are based on the identified hypotheses. As a result, the robustness of the model in a university context has been evaluated and determined whether the designed ASAG model can be deployed.

These mentioned steps together resulted in a demonstration of a high-performance ASAG model in a real-world university context and an evaluation to what extent the requirements in section 1.2 have been met.

1.5 Report Organization

The remainder of this report is organized as follows. In Chapter 2 we give the background information that provides the necessary knowledge for this thesis. Chapter 3 analyses the existing related work in the field of ASAG and provides the reasoning behind the model choices. Then, in Chapter 4 we describe the underlying methodology followed by the introduction of the used datasets. The conducted experiments and corresponding results are presented in chapter 6. This is followed by a detailed discussion of the results and limitations associated with the work. Finally, in Chapter 8, conclusions and recommendations for future work are given.

Chapter 2

Background

This chapter provides the essential background knowledge in order to follow along with the subsequent chapters. First, we introduce deep transfer learning in NLP including multi-task learning, domain adaptation, and sequential transfer learning approaches. Then, we explain the functionality of transformers as a type of neural network architecture. Afterward we will give a detailed description of the text-to-text-transfer transformer (T5) model and the used multi-task training approach. Finally, the evaluation metrics used in this work are explained in the context of imbalanced datasets.

2.1 Deep Transfer Learning for Natural Language Processing

In contrast to transfer learning, the traditional machine learning approach is an isolated learning approach where the model is trained to solve a single task. With this approach, no knowledge is retained or accumulated. As a consequence, the learning approach relies only on the single task. Whereas transfer learning as a subarea of machine learning can be described as the ability of a model to leverage learned knowledge from prior tasks to a new and unknown task. The main idea behind is that a extensively trained base model can be used for a new tasks. This makes a model training from scratch unnecessary and knowledge retainable. Eventually, this leads to a faster learning process and a generally stronger model that requires relatively less training data for good results.

Within the field of transfer learning and more specifically in NLP one has different possibilities to apply transfer learning. For this, [1] introduced a scenario-based taxonomy to differentiate between transfer learning categories which are illustrated in figure 2.1. According to [1] the different transfer learning scenarios can be arranged into two categories: *Transductive* and *inductive* transfer learning. The difference is that transductive transfer learning include methods where the source and target tasks are the same (e.g. domain adaptation and cross-lingual learning). Whereas in an inductive transfer learning setting (e.g. multi-task learning and sequential transfer learning) the tasks differ.

In this thesis, we combine sequential transfer learning in the different stages with multi-task learning and domain adaptation. Therefore, only these methods will be explained in

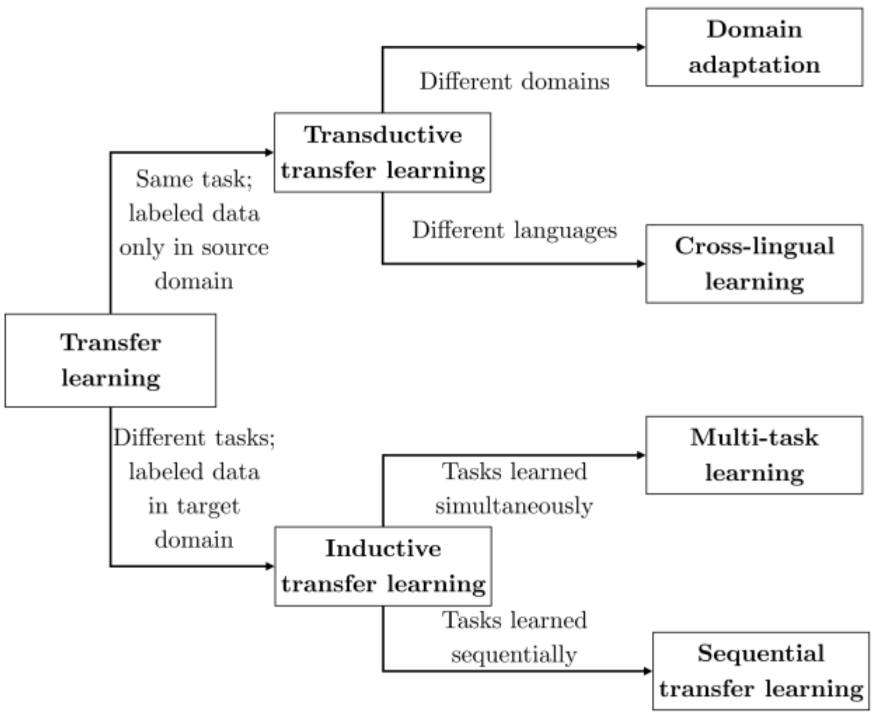

Domain adaptation
Different domains
Transductive transfer learning
Same task; labeled data only in source domain
Different languages
Cross-lingual learning
Transfer learning
Different tasks; labeled data in target domain
Multi-task learning
Tasks learned simultaneously
Inductive transfer learning
Tasks learned sequentially
Sequential transfer learning

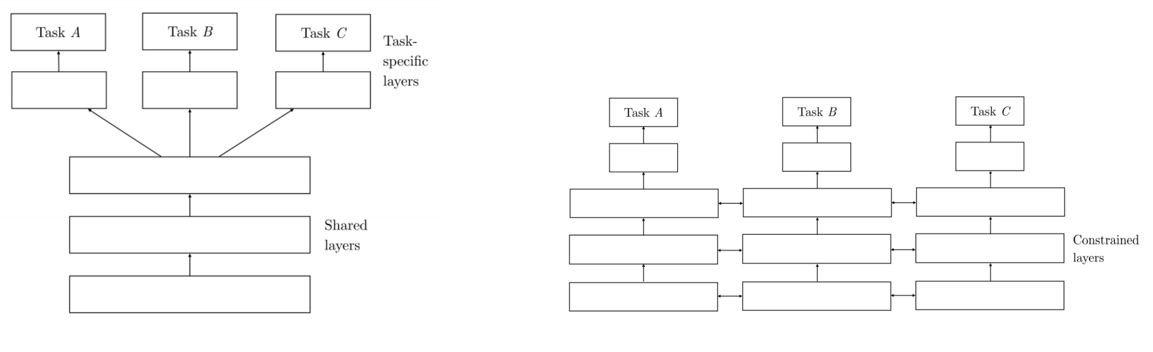
Figure 2.1: Taxonomy for transfer learning for NLP (from [1])

detail.

2.1.1 Multi-task Learning in Neural Networks

In recent years, multi-task learning (MTL) approaches have become increasingly important. Main reason for this was the good performance in various machine learning areas such as NLP [6], speech recognition [7], and computer vision [8]. The term multi-task learning describes the use of different, similar or related tasks to solve a problem by transferring knowledge gained from one task to another. In general, one can speak of multi-task learning as soon as more than one loss function is optimized. Multi-task learning has its motivation from the learning behavior of humans. Where the idea is that when learning a new task one actually applies the previously gained knowledge from other related tasks. This logic is applicable to machine learning where such a training method can lead to better performance and generalizations of the model [9].

In the following we discuss the main methods for MTL, followed by the importance of selected tasks selection and sampling strategy. Finally, we explain the associated benefits and for which problems it is useful.



(a) Hard Parameter Sharing *(b)* Soft Parameter Sharing

Figure 2.2: Methods for multi-task learning in neural networks (from [1])

Methods for Multi-Task Learning

Within the field of multi-task learning, it generally is distinguished between two different methods: Hard and soft parameter sharing between hidden layers [1].

Hard Parameter Sharing This method is one of the more popular used methods in neural networks. In such a setup the model shares several layers between the tasks and simultaneously separating task-specific output layers. This is illustrated in figure2.2 a). Since most of the layers are shared the risk of overfitting can be significantly reduced. The reason behind this is intuitive since the more tasks the model needs to learn at the same time, the more it captures diversified representations rather than task-specific [1]. This makes the training especially useful when similar target tasks exist.

Soft Parameter Sharing In contrast, soft parameter sharing uses different models for each task. This is illustrated in figure 2.2 b). Each of the models learns its own parameters but the the distance between the different layers of the models is regularized. This encouraged the different layers of the models to be similar. Commonly used regularization techniques are l_1 or l_2 norm.

Auxiliary Tasks and Sampling Strategies

Mulit-task learning is mainly utilised to solve different tasks simultaneously. However, it can also be used to solve only one specific task. In the latter case it is necessary to pay attention to the task selection. For this reason it is from great importance to analyze the main task and the auxiliary tasks that want to be used to improve the model performance. In doing so two questions are of essential importance that must be answered individually. On the one hand, it must be decided what auxiliary tasks to include. On the other hand, it is important to determine the task ratio the model is trained on.

Auxiliary Tasks For a MTL setup it is mostly useful to include related task. In general for NLP problems mostly tasks from areas such as speech recognition, machine translation, multilingual task, language grounding, semantic parsing question answering, information retrieval and more are selected. In order to decide if a task promises an advantage depends on the main task itself and is decided individually. Such a task filtering process is especially important when working with limited computational resources. In such cases a minimization of the tasks helps mitigating these problems by using a more profound dataset selection strategy.

Sampling Strategies There are different approaches for considering a task ratio which have to be chosen according to the overall goal. In most multi-task learning cases the task-individual loss functions are summed up and the corresponding mean represents the loss on the basis of which the model is updated. Therefore, one possible sampling strategy is to determine a task-specific weight factor to influence the training in favor of several tasks. As an alternative, a sample strategy can be selected accoring to a pre-determined probability distribution over the tasks. An accurate sampling strategy becomes especially important when dealing with task imbalances.

Benefits of Multi-Task Learning

Multi-task learning is connected to several advantages. One of the biggest advantage of MTL is when dealing with sparse data availability. In such a case the data can easily be extended by including more related tasks. This does not necessarily improve the performance on the target task but it leads to a higher generalization capability of the model, since parameters are learned that solve each task in the best possible way. Furthermore, multi-task learning can help models to concentrate on the essential features and neglect unimportant ones. In addition, as a general rule it can be said that if a multi-task model performs well on many tasks one can assume that it will also perform sufficient on learning new related tasks. At the same time, the regularization reduces the risk that the model will over-fit the target data.

2.1.2 Domain Adaptation in Neural Networks

Domain Adaptation belongs to the class of transductive transfer learning and is a popular approach to align model to a certain task or domain. Main characteristics of domain adaptation is that it does not aim for a good general representation but rather for a good representation for a specific target domain.

In the literature the term domain adaptation is used in different context depending on the model learning methods (unsupervised, semi-supervised and supervised). Each of them is beneficial for different problems and depend on the data availability. Compared to supervised domain adaptation, unsupervised adaptation needs a large amount of unlabeled data in order to be an effective approach. Which makes it less applicable to the data scarcity problem in ASAG. Hence, in this thesis we will only refer to supervised domain adaptation which means that labeled data is available. Basic assumption in a supervised learning setup is that the training and test data follow the same distribution. In reality however, such an assumption can be wrong when working with inherently different (e.g. multiple domains) data. In a so called multi-domain case the training distribution between the individual domains differs which can lead to a performance drop. This is the area where domain adaptation becomes important since it aim is to adapt the training distribution to better fit the test distribution.

Domain adaptation for neural networks can be applied in two mains stages of model training. It can either be conducted in the pre-training or in the fine-tuning of the model. It further differs depending on the problem context and the amount of domains. Most cases are concerned with a single source domain. However, in this thesis we are dealing with multiple source domains. This means that training and test data is available from multiple domains. Therefore, we focus on this particular multi-source domain adaptation case.

Multi-domain problems are mostly approached by pre-training a model on enough data and fine-tuning it on one domain instead of across domains. This approach has two beneficial consequences. First, the training and test distribution is expected to be more similar which increases model performance. Second, domain-specific fine-tuning increases the richness of the representations within one domain since it reduces ambiguities in word interpretation. This makes domain adaptation an efficient approach to efficiently produce meaningful input representations for a particular task.

2.1.3 Sequential Transfer Learning

Sequential transfer learning is one of the prevalent transfer learning method in NLP due to its simple usability. It can be defined as an sequential training approach where the source and target task differ. As a consequence the model learns different tasks separately rather than jointly as in multi-task learning. In essence, the goal of sequential transfer learning is to gain knowledge on a source task and transfer this to a target task. This makes it most useful in scenarios where one is dealing with a data sparse target task or where the model needs to adapt to different tasks.

In general, sequential transfer learning consists of a pre-training and an adaption stage where the previously mentioned techniques multi-task learning and domain adaptation can be incorporated. Since this is the approach the thesis utilized we will elaborate these in greater detail.

Pre-training Stage

In this stage the goal is to learn universal representations which capture general properties of natural language. This is especially effective when there is access to a large amount of data. In pre-training there is a distinction between three different methods that differ in terms of their level of availability of labeled data. In this work we refer to them as unsupervised, semi-supervised and supervised training. The results of the pre-training be used as representation of the data.

Unsupervised learning For unsupervised learning only raw text data without labels is required which makes it easy to obtain. In recent years the term self-supervised became popular which can be considered a subset of unsupervised learning. The idea is to use the raw input data and transform it to an input-target structure. This results in self-generated targets from raw textual data. Such an approach is used when the language model incorporate next sentence predictions or predictions of masked out tokens.

Semi-supervised learning In contrast, semi-supervised learning uses the raw data to automatically generate a large amount of noisy supervised data. Main difference to self-supervised learning is that noisy labels are added and the input is not just ”reshaped”.

Supervised learning Supervised learning methods can be clearly differentiated since they only deal with manually labeled training data. This makes it the most used method in machine learning.

General Word Representation Almost all models used in NLP are using unsupervised pre-training in some way. Main reason is that general knowledge and ability to detect word dependencies are crucial for most NLP tasks. Such a knowledge can only be obtained when a model is trained on a large amount of data. This makes unsupervised pre-training the most general approach to learn expressive representations of words since it works with raw unlabeled text data which makes it scaleable. In order to obtain such word representations many different approaches are used where word embeddings have shown to be superior for most cases. Word embeddings are one possible type of word representation where words with similar meaning have a similar representation.

Adaptation Stage

The adaptation is the second stage of the sequential transfer learning. It represents the knowledge transfer from the previous source task (i.e. pre-training) to the target task. There are two ways to adapt the model to a target task: Feature extraction and fine-tuning.

Feature Extraction The first is called feature extraction where the weights from the previous models are extracted and used as representation for a different downstream task. This can include different layer combinations of the model and for neural networks we refer to these representations as word embeddings. Most used techniques are summing, averaging or concatenating different layers. Such representations can be used and applied to other models.

Fine-tuning The second way is called fine-tuning and instead of extracting the representation the process includes a further parameter updating or a change of the model architecture. This allows the user to adapt a generic pre-trained model to various tasks. In general, there are three main techniques to fine-tune a model. First, the entire model can be further trained on the target task. In this case the pre-training and fine-tuning process is the same since we back-propagate through the entire pre-trained model and update the weights. Second, the model can be updated partially by keeping some layers of the architecture and only further update some parts of the model architecture. There are also many individual approaches that differ in how many layers are updated and whether this number is dynamic or static. One such popular approach is *gradual unfreezing* where the number of layers that are updated increases over time. With the last technique the entire model architecture is frozen and additional layers are attached that will be trained on the target task.

2.2 Transformers

In this section we explain the development of transformers, introduce their concepts of and the main underlying attention mechanism.

2.2.1 Transformer and Attention Mechanism

Before transformer-based architectures became state-of-the-art for most tasks, researcher used neural approaches that enable processing sequential data by remembering the important information of a textual sequence. This era of models is mostly marked by Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) models that were especially useful in handling sequence data of different lengths. LSTM models were so to speak a successor of RNNs because they prevented the model from assigning zero weights to early inputs in the sequence [10]. As a result, models were able to capture longer relationships and represent the entire sentence or paragraph in their network and based their prediction on it [11]. A change was achieved after [2] introduced transformer-architectures and the attention mechanism. The Transformer is a novel neural architecture which is especially suited for sequence-to-sequence tasks because of the ability to capture long-term dependencies in sequential data like text [2]. The main idea behind the transformer architecture is to use so called attention mechanisms to generate word/sentence representations. The general transformer architecture is based on encoder and decoder stacks and uses attention to determine the word representation.

Encoder and Decoder Stacks

The transformer architecture is illustrated in figure 2.3. In contrast to previous architectures, transformer models completely rely on multi-head self-attention mechanisms. A transformer model consists of an encoder (left block) and a decoder (right block) part in which one or more encoder/decoder are stacked together. All the stacked encoder have an identical structure with a Self-Attention layer and a Feed Forward Neural Network followed by a layer-normalization step. In contrast the decoder contains in addition a Multi-Head Attention layer applied over the encoder output. The architecture also modified the Self-Attention layer via masking in a way that prediction only depends on the known outputs at the current position. In the following we will discuss the most important functions of this architecture the attention mechanism.

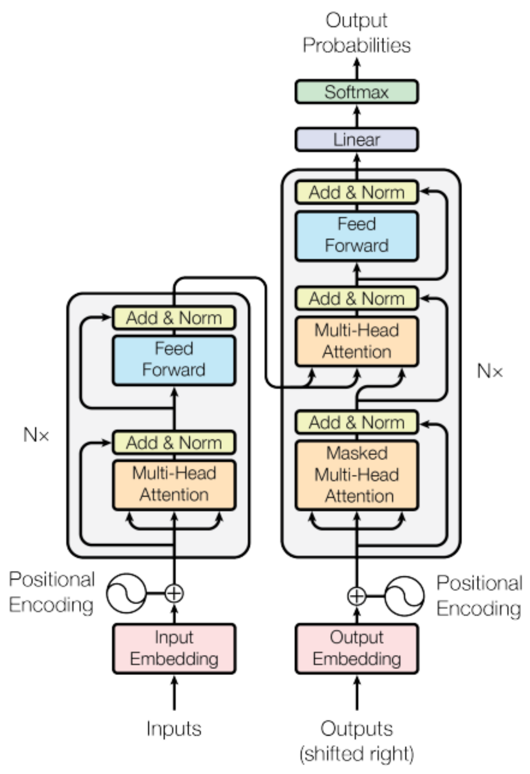


Figure 2.3: Transformer architecture (from [2])

Multi-Headed Self-Attention Mechanism

The transformer structure in figure 2.3 shows that the encoder and decoder use multi-headed self-attention. For this reason we first elaborate self-attention and then go into details with multi-head attention proposed by [2].

Self-Attention Self-attention refers to the attention between within a input sequence or also with an output sequence. The idea behind the calculation of the self-attention is that a word representation is represented by the weighted sum of the individual token inputs of the sequence. The assigned weight corresponds to a similarity measure between the target and source token. The exact underlying process with vector representations is described below.

Before calculating self-attention three vectors are created by multiplying the input word embedding with three different weight matrices. These matrices are learned during the training process. After the multiplication three different representations of each input sequence (query vector (Q), key vector (K) and value vector (V)) are obtained. In a next step, each individual word receives a attention score for each word in the sentence. I.e. each individual word is scored against the current word in the sequence. This score represents the importance or attention for a particular input. The score is calculated by taking the dot-product of the query vector (from the current word) and the key-vector of each word containing in the input sequence. For the first input one attention score for each word (including the word itself) is obtained. These scores are divided by the square root of the dimension of the key-vector and feed into a softmax function in order to obtain weights that sum up to one. This process is called Scaled Dot-Product Attention (see figure 2.4). In order to obtain the final word representation, the weighted sum of all the weights and the corresponding value-vector of the word is calculated. This new attention matrix (or embedding) for each word is so to speak a weighted combination of all the words in the input sequence including the word itself. As a result we end up with a matrix representation of each input.

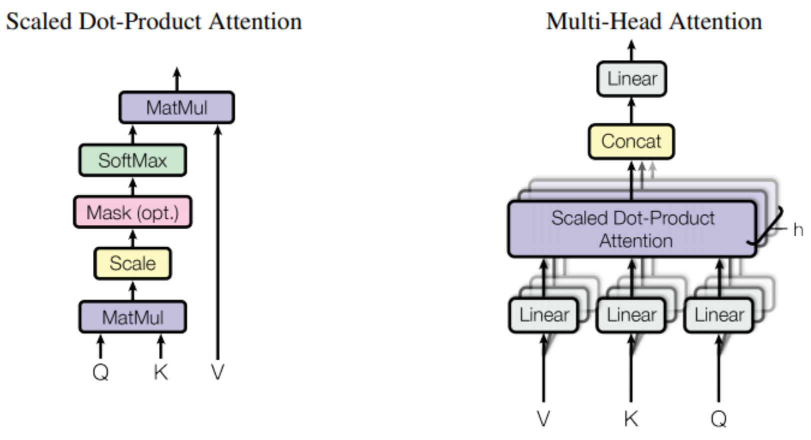


Figure 2.4: Scaled dot-product attention and multi-head attention (from [2])

Multi-Head Attention The above explained process is one possibility for self-attention and can be also considered as a one-head attention since during training only one weight matrix for the query, key and value matrices is learned. The authors' of the paper realized that in contrast to a single linear projections (weight matrix), multiple projections can be beneficial. This is also known as multi-head attention. Essentially, the same linear projection is done multiple times where the weight initialization of the query, key and value matrices are different. Dependent on the number of used heads the final representation is a concatenation of the self-attention results for each head. This is illustrated in the on the right side in figure 2.4

On the decoder side figure 2.3 we can see an attention layer where the output of the encoder stack and the input of the decoder are brought together. This so called encoder-decoder attention layer is comparable to the multi-head attention where the query matrix comes from the decoder and the key and value matrix from the encoder. Main difference is that future outputs are masked to make sure that the final prediction are only based on known outputs.

Positional encoding

Since the model does not use recurrence of convolution it is important to keep track of the order of the words in the input sequence. In the transformer architecture this can be done in different ways, but the one addressed by [2] is to add an positional encoding (i.e. simple vector) to the input embedding. Broadly speaking, by adding this positional encoding the resulting embeddings contain information about the distance between words in the input sequence. For more details about the positional encoding and the underlying functionality see [2].

Benefits of attention-based architecture

Such attention-based transformer architecture with capability to pay attention to a specific subset of the sequential input data helped improve the performance of several NLP tasks. These models became state-of-the-art for most NLP tasks due to their better and more efficient performance in terms of computational resources. As described the most popular type of attention-based networks are the transformers which handle sequential data simultaneously rather than just sequentially like RNNs [10]. This lead on the one hand to a faster and more extensive model training and on the other hand to an increased usage of transfer learning of such pre-trained models [12].

2.3 T5 Model Architecture

The research paper published by the authors' gives an overview of different transfer learning methods and introduces a novel approach to combine any natural language tasks. The proposed method transforms natural language tasks into a text-to-text format. By doing so one model can be trained on several tasks simultaneously. This flexibility in the integration of different tasks enables a T5 model to be used in an enormously wide range of applications and reduces the need for individually task-specific trained models. In their work the authors' carried out many experiments that they combined in their survey paper. We will not summarize the content of this paper as they can easily be found in [3] and various other sources [13] [14]. We only explain and discuss the final and most suitable result of the paper that results in their published trained model. For detailed explanations and the different approaches investigated we refer to the paper [3].

In the following, we will explain first the new text-to-text format and the unique input and output representation of the model. This is followed by an introduction to the underlying C4 dataset. Afterward we describe the model architecture and the used training approach for the model.

2.3.1 Text-to-Text format and Input/Output Representation

The novel unified framework which allows the model to combine all language problems in a text-to-text format in one model is the core of the T5 model. The systematic behind it can be seen in the following picture.

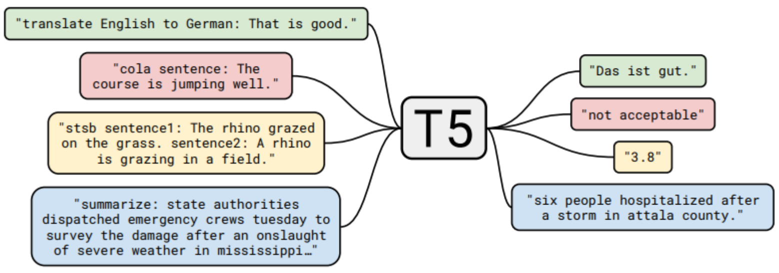

"translate English to German: That is good."
"cola sentence: The course is jumping well."
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"
T5
"Das ist gut."
"not acceptable"
"3.8"
"six people hospitalized after a storm in attala county."

Figure 2.5: Diagram of the text-to-text framework (from [3])

As the name of the model *text-to-text transfer transformer* implies the main idea is that it treats each NLP task as a ”text-to-text” problem. In detail, the model receives as input a simple string and produces a string output (i.e. text output). In order to distinguish between the different tasks, a unique prefix is added to each input sequence from a task. This approach is based on the assumption that the model learns to recognize each task by its prefix and outputs in intended labels in a text version. This framework allows to use a single model, with a single - although combined - loss function including all NLP tasks. This makes it an unique multi-task learning approach since all model parameters are shared between tasks and the model simply learns to predict different labels according to the added task prefix. Figure 2.5 illustrates the T5 model framework where it combines different NLP tasks like machine translation, similarity tasks and summarization. Even regression tasks can be used by not predicting a continuous variable but rather consider the string representation of the variable as a single label class. This differentiation between tasks however is associated with the risk, that the model makes predictions that do not correspond to the intended labels of the task. In such cases, the model is trained to interpret deviating predictions as a wrong. However, according to the authors', this never occurred in their experiments which indicates that the model indeed learns to differentiate between the different tasks.

2.3.2 C4 - Colossal Clean Crawled Corpus

In their paper, the authors' pursued the goal of analyzing to what extent the up-scaling of pre-training has an impact on performance. Therefore an enormously large and diverse dataset was needed. For this reason, the authors' developed a dataset called Colossal Clean Crawled Corpus (C4). The final dataset contains 750GB of clean english text scraped from the web. It was created with a month of data from the common crawl corpus cleaned with a set of filters that filtered out “bad/useless” text (e.g. offensive language, source code, etc.). As a comparison and to illustrate the enormous size of the dataset, models like BERT

[15] used only 13GB of data for training and XLNet [16] 126GB.

2.3.3 Model Architecture

In the following the key points of the model architecture are mentioned, as they led to the chosen model and training architecture used in this thesis.

The T5 model architecture is aligned to the described encoder-decoder transformer implementation proposed by [2] which is explained in section 2.2. The general process can be described in several steps that reflects the model architecture. First, the model learns with a SentencePiece tokenizer [17] how to encode the WordPieces tokens [18] [19]. In a next step, these sequences of tokens are transformed into an embedding and passed to the encoder. These embeddings have 1024 dimensions which are the same as for each sub-layer. The Baseline T5 architecture also works with a stack of encoders where each consists of a self-attention layer and a feed-forward layer. Each of the feed forwards layers have an output dimension of 3072 and use ReLU activation function. As dimension for the key and value matrices, 64 was chosen with 12 different attention heads. The encoder and the decoder consist of 12 blocks. In addition layer normalization [20] is applied but only the activation is re-scaled without applying additive bias. This is followed by a residual skip connection [21]. Furthermore, a dropout probability of 0.1 [22] is applied (on the feed-forward network, attention weights, skip connection, and input/output of stack). The decoder structure is the same as described in 2.2.1 where in contrast to the encoder it uses causal self-attention in order to prevent that the encoder attends to future outputs. The final decoders output is fed into a dense layer (weights are shared with the input embedding) and a softmax function is applied. Furthermore, the model uses also Multi-Head Attention and in contrast to the proposed model from [2] they used relative position representation [23] [24]. In the paper, it was proven that up-scaling the model size leads to an increase in performance. For this reason, the authors' trained models of different sizes, the specification and their results of the two biggest models are shown here.

”**3B and 11B Model**: For both model they use d_{model} [a] = 1024, a 24 layer encoder and decoder, and d_{kv} [b] = 128. For the “3B” model, they used d_{ff} = 16,384 with 32-headed attention, which results in around 2.8 billion parameters; for the “11B” model they used d_{ff} [c] = 65,536 with 128-headed attention producing a model with about 11 billion parameters.” [3]

[a] d_{model} = Sub-layers and embedding dimensions
[b] d_{kv} = Key and value matrix dimensionality of all attention mechanisms
[c] d_{ff} = Output dimension of Feed-Forward layer

2.3.4 Unsupervised Training Objective

As the training objective for the unsupervised task the model uses BERT Masked Language Modeling. The model masked out 15% of the tokens where the target is to reconstruct the uniquely masked out tokens. In contrast to BERT, the T5 model replaces tokens with a range of masked out tokens (e.g. $< X >$, $< Y >$ and $< Z >$). Furthermore, consecutive tokens (i.e. span) are replaced by only one token. This schematic can be seen in figure 2.6.

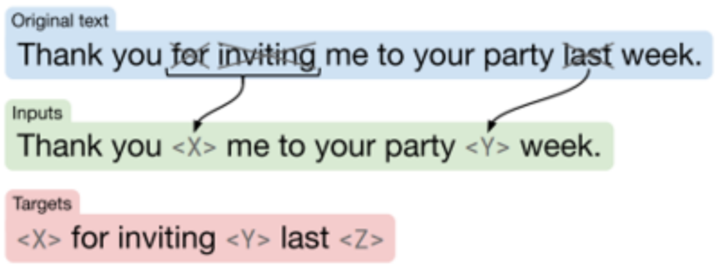

Original text
Thank you ~~for inviting~~ me to your party ~~last~~ week.
Inputs
Thank you <X> me to your party <Y> week.
Targets
<X> for inviting <Y> last <Z>

Figure 2.6: Schematic of the unsupervised training objective (from [3])

2.3.5 Training Strategy

The training strategy is divided into two parts. The multi-task pre-training and the subsequent fine-tuning of the model for the respective downstream task.

Multi-task Training

The model uses a multi-task learning approach in which it combines the previously mentioned unsupervised and several supervised NLP task. In total it used all datasets from the GLUE, SuperGLUE, WMT, CNN/DM and SQuAD tasks which amount to 23 different NLP tasks. The authors' refine the multi-task learning term by simply mixing dataset together. An important point in such multi-task training is the mixing ratio between the datasets in a particular batch. As a task mixing ratio, the models uses an approach called example-proportional mixing which helps with large imbalances between the datasets. This procedure selects samples according to the respective dataset distribution. However, since the C4 corpus is disproportionately larger a dataset size limit implemented. Such a limit is used to calculate the probability of drawing a sample from a specific task.

The text-based pre-processed input allows using teacher forcing for standard maximum likelihood training. Since this model architecture requires the prediction of a sequence it produces a probability distributions over each possible output. To decode the sequence all possible output sequences (corresponds to the target labels from the task) are searched based on their likelihood. To approximate the sequence with the highest probability at each time step greedy decoding is used. As hyper-parameters during pre-training the model uses AdaFactor optimizer [25] and a learning rate schedule defined as $1/\sqrt{max(n, k}$ with *n*=current training iteration and *k*=number of warm-up steps. This means during warm-up the learning rate is constantly 0.01 and decays after exponentially. The model is trained for 1,000,000 steps with a batch size of 2^{11}.

Fine-tuning

For fine-tuning, the model uses the method of updating all pre-trained layers when training on downstream tasks. In contrast to the model training while fine-tuning the model uses a constant learning rate of 0.001. For further information about technicality please see the original paper [3].

2.4 Evaluation for Imbalanced Dataset

In this section, we describe the used evaluation metrics and elaborate the specifics and benefits. Evaluation of the trained model is a crucial part of deploying a machine learning model. A common problem with evaluating the performance of a machine learning model is to choose the right metrics. In order to give an overview of the different metrics for classification problems, we explain the main metrics and their relevance for binary classification and multi-class classification.

2.4.1 Binary-Class Evaluation Metrics

In general accuracy is mostly used as metrics for performance evaluation. However, in some cases it is not enough to reliably evaluate the model performance. One example is the case when one is dealing with multi-class dataset with an imbalance class distribution. In such a case a model could achieve a high accuracy by simply predicting the maturity class all the time. Since this can be misleading and makes the model impractical other ways of performance evaluation can be used.

One of the main metrics for model evaluations are parts of the confusion matrix. In general the confusion matrix visualizes the model predictions and the true sample class of a prediction. This is a way of visualizing the model performance for each class. One of the main benefits is that several metrics can be derived from the confusion matrix that are from great relevance. For the sack of understanding we consider a binary classification problem where a student answer is either correct (positive) or false (negative). The main elements of the confusion matrix are illustrated in figure 2.7 and are defined as follows:

- True positive (TP): The value represents the number of student answers that are actually correct (positive) and classified as correct (positive).
- False negative (FN): The value represents the number of student answers that are actually correct (positive) and classified as false (negative).
- False positive (FP): The value represents the number of student answers that are actually false (negative) and classified as correct (positive).
- True negative (TN): Its value represents the number of student answers that are actually false (negative) and classified as false (negative)

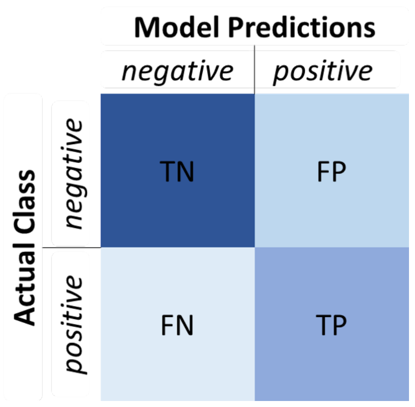


Figure 2.7: Example of a confusion matrix

Further important evaluation metrics for the model performance can be derived or determined from the confusion matrix. These are illustrated in table 2.1. Each of the metrics measures a different property of the classifier which leads to trade-offs between metrics such as precision and recall.

Table 2.1: Overview evaluation metrics for measuring model performance

Metrics	Formula
Accuracy	$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$
Precision	$Precision = \frac{TP}{TP+FP}$
Recall or Sensitivity	$Recall = \frac{TP}{TP+FN}$
Specificity or True Negative Rate (TNR)	$Specificity = \frac{TN}{FP+TN}$
F1-Score	$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall}$

The goal for a good classifier is to achieve high precision and simultaneously a high recall value. Meaning that there are no false-positive or false-negatives. Since there is a trade-off between precision and recall the f1-score is used to express these two metrics in a single metric. The F1-score is computed by the formula:

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{2.1}$$

By using the harmonic mean the F1-score makes sure that a low score becomes a large weight. Meaning that, for instance, in case a classifier achieves a precision of 100% whereas the recall is 0%, the f1-score will not be the arithmetically mean (50%) but 0%.

2.4.2 Multi-Class Evaluation Metrics

Similar to the binary case precision and recall can be calculated for each class in a multi-classification problem.

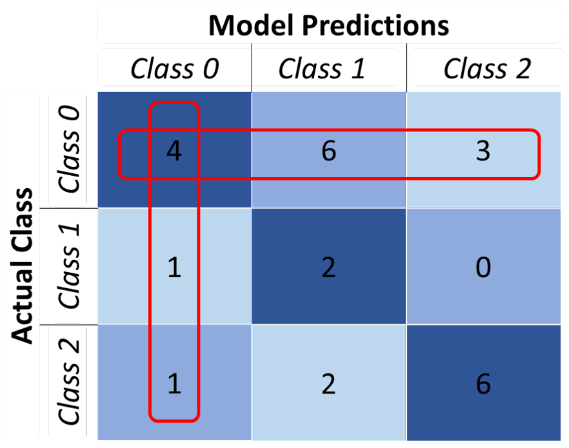


Figure 2.8: Example of a multi-class confusion matrix

For the example in figure 2.8 the precision of class 0 is given by the number of correctly predicted instances divided by all predictions of class 0. Therefore the precision results as follows.

$$Precision = \frac{TP}{TP + FP} = \frac{4}{4 + (1 + 1)} \tag{2.2}$$

Whereas the recall is given by the number of correctly predicted instances divided by all actual class 0 instances. This is illustrated in the following formula.

$$Recall = \frac{TP}{TP + FN} = \frac{4}{4 + (6 + 3)} \tag{2.3}$$

With this approach, the precision and recall for each class can be calculated. After applying equation 2.1 for each class and obtain for each class one F1-score. To measure the performance of a model with a single metrics on a downstream task different f1 related metrics can be used. The simplest one is the macro-average f1-score that calculates the arithmetic mean of the f1-scores for each class *i*:

$$F_1^{macro-average} = \frac{\sum_{i=1}^{N} F_1^i}{N} \tag{2.4}$$

When using the macro-average f1-score we treat each class the same which fails to reflect the true performance of a classifier in an imbalanced dataset since minority classes are considered the same as majority classes. To avoid this and to consider imbalances in the dataset the weighted-average f1-score is used since it weighs each class by the number of samples. It is defined by:

$$F_1^{weighted-average} = \sum_{i=1}^{N} w_i * F_1^i \tag{2.5}$$

For the evaluation of the models implemented in the thesis, the key figures weighted average f1-score, macro average f1-score, precision, recall are used.

Chapter 3

Related Work

This chapter reviews the most relevant work related to Automatic Short Answer Grading. In particular, we discuss the latest advancements regarding ASAG and their chosen model architectures. Then we introduce different ASAG datasets and derive the current state-of-the-art baseline. This is followed by a short literature review on commonly used methods in the fields of multi-task learning, domain adaptation and model explainability. We then conclude the review by explaining the selection of the models used in this thesis.

3.1 Automatic Short Answer Grading

Existing approaches in the field of ASAG fall into two broad categories. (1) Early approaches which rely on hand-crafted features. (2) Deep learning approaches which minimized manual feature engineering and focus on the semantic information [26]. The latter category can be further divided into three subcategories that align with the advancements in Natural Language Processing. The focus of this section is exclusively on deep learning models.

3.1.1 Deep Learning Approaches in ASAG

The models found in the literature dealing with short answer grading align with the general advancements in natural language processing. The analyzed models can be divided into three categories that correspond to the historical development of NLP. First, models that made use of the advancement of Word Embeddings with the Word2Vec breakthrough. Such models mostly use word-representations and sentence embeddings. Where latter ones are generated by summing or averaging the single word embeddings. These representations are used to generate more advanced features which contain more semantic information [10]. Second, neural approaches that enable processing sequential data by remembering the important information of a textual sequence. This era of models is mostly marked by Recurrent Neural Networks (RNN) Long Short-Term Memory models that were especially useful in handling sequence data of different lengths. LSTM models were so to speak a successor of RNNs because they prevented the model from assigning zero weights to early inputs in the sequence [10]. As a result, models were able to capture longer relationships and represent the entire sentence or paragraph in their network and based their prediction on it [11]. Third, attention-based neural network with the capability to pay attention to a specific subset of the sequential input data. These models became state-of-the-art for most NLP tasks due to their better and more efficient performance in terms of computational resources. The most popular types of attention-based networks are the transformers that can handle sequential data simultaneously rather than just sequentially like RNN's [10]. This leads on the one hand to a faster and more extensive model training and on the other hand to an increased usage of transfer learning of such pre-trained models [12]. These three categories are elaborated in greater detail in the following with a focus on their specific benefits and drawback on ASAG.

With advances in deep learning, new approaches for short answer grading have become more popular. Most of them were using word embeddings [27] [28] [29]. The main reason for this was the advantage in their semantic richness of the embedding representations. The review methods made use of word embedding in order to enhance the calculated similarity between words from the student answer and from the reference answer. Nevertheless, the researchers emphasized that simple word embeddings are beneficial for ASAG for calculating word similarity but lack in their performance when used as sentence representations. [28] for instance used different types of similarity measures based on word vector representation obtained by pre-trained word embedding models (Word2Vec and GloVe). By using features based on word and sentence embeddings they achieved below-average results on the SemEval 5-way task. Other approaches that solely focus on features extracted from embeddings did not outperform the models with more engineered features. Therefore, most of the initial attempts are characterized by trying to compensate for the limitations of word embeddings by extracting additional engineered features. One method proposed by [27] was to incorporate external knowledge like paraphrase database and WordNet and syntactical similarity features. With their approach, they achieved one of the best scores on the SciEntsBank dataset for unseen answers. The research further indicated that using word embeddings increased the model's generalizability. This can be seen in their performance on the SemEval task when handling unseen questions and domains where they increased the performance of the models. With further development in deep learning researchers used in addition sentence representations obtained by summing or averaging the word embeddings in a sentence. [29] for instance used sentences and word alignments by calculating lexical and semantic similarity. Lexical similar word-pairs are obtained by using an external paraphrase database. Further, he calculated the semantic similarity (cosine similarity) with word vectors extracted from pre-trained word embedding models and represented a sentence by summing the corresponding word embeddings. Their approach resulted in relatively good results which indicated the usefulness of their feature selection but also emphasized the necessary future developments of sentence representations.

Nevertheless, the used embeddings had limitations in usability and a significant increase of different neural model architectures was no longer visible according to the review. Therefore, researchers like [11] investigated the importance of domain adaption of a short answer grading system. They argued that systems that rely on textual similarity, paraphrasing or entailment can be dependent on the domain. He analyzed this by proposing a joint multi-domain deep learning architecture that uses domain adaptation. The system uses an encoder with a BiLSTM layer to transform the reference answer and the student answer into an output embedding. Based on the result, a similarity scorer is trained domain-specifically by using the just in-domain training data. In order to represent additional generic features, the system also has a generic similarity scorer which was trained on all the domains. For the final prediction, the system adds the domain-specific with the generics class scores. Their approach achieved the best overall results on the SemEval task and therefore showed that a combination of in- and off-domain training combined with word/sentence embeddings can improve the performance. Unfortunately, the overall best model reported only the mean score for the three categories and not for each category separately. This makes the interpretation of the model approach difficult since it is not clear how the model performed in the different categories.

Other subsequent research extended these approaches by exploring the effect of more sophisticated feature representation methods in order to capture more semantic information and structural information. [12] investigated how neural network approaches perform on short answer scoring. They used different basic neural networks like LSTM, CNN and attention mechanism. In their research, they showed that neural architectures can outperform previous methods. Especially bidirectional LSTMs and attention produced good results. However, they empathized the importance of the used input embedding and the parameter setting. According to the author it is crucial to fine-tune pre-trained models for the tasks in order to obtain good results. In addition, they stated that parameter optimization and the combination of different neural architectures may result in a better performance. Most of the followed approaches made use of transfer learning with the goal to generate more contextualized embeddings by using different architectures and different attention mechanisms.

A change was achieved after transformer-architectures were introduced [2]. These allowed to create better word and especially sentence representations. Based on that development further research was focusing on exploring the effects of using embeddings which are based on transformer-based architectures with an attention mechanism. [26] showed with his research the superiority of such deep learning approaches by leveraging a fine-tuned Bidirectional Encoder Representation from Transformers (BERT) architecture for short answer grading task. The author showed that transformer-based pre-trained models can produce superior results in ASAG. However, their research also showed the domain dependency of trained models and the resulted lack of off-domain generalizability. In addition, their research indicated that task-specific transfer learning with BERT can be beneficial even without many training samples. As a continuation of [26] [30] investigated the improvement potential of the contextual representation of the sentences. They showed that by utilizing unstructured domain text data and question answer pairs better results compared to a simple task-specific fine-tuning can be achieved. Those developments highlight the importance of transfer learning and domain-specific training. With their approach they were able to outperform all the previous work on the SemEval 3-way task which indicates the superiority of the model. Unfortunately, they did not report the results on the 5-way task which makes the comparison more difficult.

3.1.2 Datasets for ASAG

Most of the related research papers dealt with a great variety of datasets. These datasets have many differences regarding the number of questions, the number of answers, question type, domain, language, grading basis, answer length. To the best of our knowledge, 12 frequently used different datasets have been identified in the review. Some of the models are evaluated on more than one dataset, but there is no clear standard in terms of comparability because of different characteristics and properties. In addition, some of the used datasets were connected with uncertainty and did not provide a reference. Due to inconsistencies in references and the unavailability of the dataset we filtered the following four most commonly used datasets and analyzed their usability:

1. SciEntsBank dataset as part of the SemEval 2013 Task [31]
2. Beetle dataset as part of the SemEval 2013 Task [31]
3. University of North Texas dataset [32]
4. Automated Student Assessment Prize Short Answer Scoring (ASAP-SAS) dataset [33]

In general, all datasets are suitable for comparing different short answer grading systems. The ASAP-SAS dataset mirrors the real world and the diversity in exam questions quite well. Nevertheless, the diversity of the task causes problems in finding a general approach for all the questions and focusing on a single question would decrease the data disproportionately. Furthermore, the ASAP-SAS and the University of North Texas dataset used regression for the score prediction. Whereas the SciEntsBank/Beetle dataset uses exactly 5 predefined categories which also indicate missing parts in the student answer. This results in the possibility to include feedback which is closer to a digital tutor. In addition, the SciEntsBank task seems to be a more realistic and challenging test for an automated tutoring system. Within the SemEval task the SciEntsBank dataset encompasses several domains that makes it more suitable for the analysis of domain adaptation possibilities than the Beetle dataset. For these mentioned reasons, in this thesis, we will focus on the SciEntBank dataset.

Dataset Baseline and Model Evaluation

As part of the previously conducted review, we analyzed the model performances on the target dataset. One of the results was the comparison of the best published models and their performance on the SciEntsBank dataset. Based on this the best performing models on the SciEntsBank 5-way classification task are collected and determine our baseline. Due to the difficulty and the inconsistencies in reporting results, the baseline is defined in a more meaningful way. Most research defines the success of the model by reporting average scores of the three categories Unseen Answer, Unseen Question and Unseen Domain rather than reporting scores for each category separately. In order to compare our model in a more informative way, we build the baseline according to the best achieved scores from different research models for each category. This baseline and the best performing models for each category can be seen in table 3.1[1][2].

Table 3.1: Baseline for the SciEntsBank dataset using weighted average f1-score

Model	Average**	Unseen Answer*	Unseen Question*	Unseen Domain*
[11]	0.6565	-	-	-
[27]	0.5656	0.6720	0.5180	0.5070
[34]	0.5903	0.6660	0.5310	0.5740
Baseline	**0.6565**	**0.6720**	**0.5310**	**0.5740**

3.2 Multi-task Training

In this review we analyses current multi-task models that can be used for the given problem context of ASAG. The latest advancements and approaches on mulit-task learning model methods are analyzed extensively in [35] and [36]. We limit the analyzed models to the field of NLP.

Commonly used approaches selected individual training methods for different layers and shared the embeddings accordingly. The underlying idea is that simpler related tasks are trained in earlier layers and can be used in subsequent layers for more advanced tasks. [37] for instance used a *joint many-task model* that uses the linguistic hierarchies to train a neural network for an entailment task. The idea behind it is that the model needs to perform well on low-level tasks (e.g. POS tagging) before training on more complicated tasks. Such training approaches are especially useful when linguistic features are important for solving the target task. [38] used a similar approach for coreference resolution and relation extraction. In contrast to [37], they used a shortcut connection between the multi-layer encoder modules for each task rather than optimizing all weights in the training. This means information from lower-level tasks is shared by concatenating their embeddings with upper-level embeddings. With this approach, they prevent the model of so called catastrophic forgetting [39] where the model forgets knowledge from previously trained tasks. Other approaches like [40] achieved good results for learning sentence representations across tasks. They used a hard parameter sharing approach where lower layers are shared and top layers are trained task-specific.

Another line of research focused less on the model architecture and task hierarchy but more on the ability to easily integrate tasks into a combined model by transforming the input and output data. [41] predicted one target sequence which consisted of several tasks. In their attention-based encoder-decoder model they transformed machine translation, POS

[1]* Results are reported as weighted average f1-score
[2]** Average is calculated by taking the mean (not weighted) over the three categories

tags and NER tasks into a single prediction problem. This means that the target sequence is a concatenation of the translated input, the corresponding POS tags and the named entities of the target sequence. They concluded that a simultaneous training of different tasks in one sequence is not preferable and splitting can be beneficial. A similar approach for improving word representation was used by [42]. They incorporated machine translation, constituency parsing and NLI tasks. The authors' transformed the tasks into a sequence-to-sequence problem. This choice enabled the model to train a single shared encoder and task-specific decoder. The authors' showed that by sharing the encoder across related tasks substantial improvements can be achieved.

Other researchers followed a sequence-to-sequence approach where different tasks are transformed and combined in one encoder-decoder architecture. [43] trained one single translation model jointly on dependency parsing and part-of-speech tagging. They used task-specific embeddings so the model recognizes the different tasks during training. This allows the model to be trained efficiently and to adjust the sample ratio of the tasks individually and easily. Another method that achieves very good results on different tasks is [3]. In contrast to most of the models, their approach focuses on adapting the input format of the datasets. In their work, they introduced a text-to-text architecture which made it even easier to include different tasks without changing the model architecture. This makes the model highly flexible is usability and easily adaptable to new tasks and.

3.3 Domain Adaptation

In this review, we focus on domain adaptation in the pre-training and adaptation (i.e. fine-tuning) stage for different NLP tasks. Based on these findings we elaborate our domain adaptation approach used in this thesis.

With the development of excessively trained neural models researchers analyzed if it is beneficial to pre-train model on particular domains. [44] showed in a comprehensive study that a domain-adaptive pre-training of a language model (RoBERTa) is beneficial. Other approaches achieved similar results. For instance, the BERT model was further successfully pre-trained on literature of biomedical and computer science [45], financial services [46] or patent data [47]. However, this requires a large amount of domain-specific data which causes trouble in data collection and therefore makes it particular difficult to apply without access to domain-specific data.

To circumvent this problem while take advantage of the benefits focused on the fine-tuning process on a particular domain. Such approaches proofed beneficial when the target data does not encompass many training samples and one wants to compensate for such low-resource data. [48] illustrated in their work that a combination of pre-training on one followed by fine-tune on another domain can be beneficial for data spares tasks. The authors' concluded that such a setup improves the generalization ability of the model and still keeps the model focused on the particular target domain. [49] showed similar results by pre-training a model on generic text followed by domain-specific fine-tuning on the target task.

Other approaches indicated that a multi-stage fine-tune process can increase performance for extracting definitions from free text. [50] compared different fine-tune strategies and found that fine-tuning a model twice can be beneficial. In their setup the authors' first fine-tuned the BERT based model on the full dataset and then fine-tuned it further towards the domain-specific data. Other modification of the fine-tuning process were used by [51] for a translation task. The authors' used a multi-stage fine-tune process on different compilations of datasets and showed that heir method outperformed normal single-stage fine-tuning. However, this is associated with significantly increased computational costs since they fine-tune the model multiple times.

3.4 Model Explainability and Interpretability

In NLP there is a trend in the last years that research use bigger and more complex models that are trained with an enormous amount of data and computational resources. Despite the significant performance improvements, these models raised questions regarding their interpretability and explainability. Within this section, we review the latest research focusing on the interpretability and robustness of NLP models. The research in this field can be roughly distinguished in three categories: Direct model understanding, Model-agnostic approaches and example-based explanations.

3.4.1 Direct Model Understanding

For a better understanding of the internals of the model, it is common to use the attention mechanism (i.e. attention weights). Underlying assumptions for its usability is that since attention layers weigh the inputs and generate embeddings it can be used to identify particular important tokens ([52], [53]). However, it is not clear if there is a relationship between attention weights and the actual model prediction. Several researchers doubted the usability ([54], [55]). [54] for instance stated that only in some cases attention weights actually correlate with model predictions and that it can not be assumed that it would hold in most cases. Other provided several methods were the attention weights could be successfully used for inferring model decision ([56], [57], [58]).

3.4.2 Model-agnostic Approaches

Since the internal of such a complex model is difficult to understand, separating the model from the explanation was found to be advantageous. This is addressed by model-agnostic interpenetration methods which simplify the interpretation by focusing on the model predictions rather than the internals of the model.

One popular method is called LIME (local interpretable model-agnostic explanations) [59]. Lime builds a local surrogate model that explains particular instances of the model. The methods train a linear model on different generated samples (i.e. manipulations of one sample) of an instance to predict the original model predictions. With this surrogate model, important instances for a prediction can be identified. Due to its local focus it is, however not clear if the explanation is also applicable for unseen instances. Furthermore, simple surrogate models are restricted in their learning and can not fully capture the prediction behavior which results in an oversimplification of reality.

Other researchers extend the mere local focus of the LIME model. [60] introduced a concept called anchors that identifies tokens or sequences (i.e. anchors) which are sufficient for a certain model prediction. These anchors are found by including the entire datasets rather than only one single sample. That makes it possible to derive more general prediction rules that are also applicable for unseen instances. However, this is associated with a considerable computational effort depending on the dataset size.

The L2X Model proposed by [61] is another possibility to extract important features within a language task. The idea is that it selects for each instance separately the most important feature (i.e. token). The method is based on a CNN architecture that learns to extract *n* numbers of tokens from an input sequence that are most informative. A model is trained as a feature selector that selects a subset of words based on their mutual information between the token and the target variable. This model is especially useful when one is looking for fast and efficient identification of local features since it does not require a huge amount of evaluations of the re-sampled original model inputs like LIME.

3.4.3 Example-based Explanations

Another approach to explain a prediction distribution is to generate example-based explanations that are based on findings from model-agnostic methods. With this method, one tries to draw conclusions about the model behavior by generating samples and test them on the respective model. One line of research focuses especially on adversarial examples as they simultaneously explain and test the vulnerability of the model. For detailed information, we refer to a comprehensive survey about adversarial attacks from [62].

Broadly speaking such adversarial attacks for NLP can be structured in gradient-based ([63], [64]) or generative-based approaches which also include manual sample generations. [65] used the gradients of the input representation to determine the best possible input manipulation that increases the model's loss function. [66] and [67] followed the approach but generated input-independent token sequences that trigger the model to classify the opposite label. With such gradient-based approaches one mostly exposes superficial patterns which are efficient in tricking the system but fails to consider realistic sequences. For this reason, other approaches have been developed that aim to transform a sample by preserving the original meaning ([68], [69]). This improves the sample quality but still fails to capture the diversity which is expected in reality. Therefore, such approaches are solely beneficial to improve the model but are limited in their diversity and usefulness for robustness tests for ASAG models. [70] addressed this issue and used professionals to generate adversarial samples. Such manual approaches cover more diverse patterns resulting in complex and diversified examples. This is however achieved at the expense of scalability.

Another line of research focuses on specific behavioral tests of the model to infer model capabilities. Most approaches aim for evaluating the machine comprehension of textual data. [71] proposed a framework called CheckList to facilitate behavioral testing of linguistic capabilities supplemented with individual test types. Others evaluating specific behaviors which makes them only applicable for particular behaviors. For instance [72] and [73] targeted the solely on evaluating the model on detecting grammatical errors.

3.5 Conclusion Review

Based on the previous review we decided which model is most useful for the different requirements with a focus on model architecture, fine-tuning approach and model explainability, and model robustness.

A suitable model architecture for the given problem needs to be first and foremost simple and efficient to use. In particular, it is necessary to deal with the multi-domain SciEntsBank datasets and to provide the possibility to easily incorporate different NLP tasks. The T5 model architecture is best suited for these requirements due to its flexibility and proven high performance. In contrast to other models, T5 is a well studied and optimized model with unique characteristics in applying multi-task learning. This makes it particularly easy to incorporate different structured datasets. In terms of domain adaptation it is most useful to apply domain-specific fine-tuning with the model choice.

Due to the enormous size of the T5 model and the associated ambiguities in the usability of the attention weights, it does not seem promising to concentrate on the internal functioning (e.g. attention weights) of the model. Instead, model agnostic approaches are more suitable. For the ASAG area, models that focus on local features are particularly useful. This is due to the fact that questions and student answers are generally very different which makes identification of general patterns of limited use. Furthermore, the algorithm should be easy to implement and efficient to use. This makes the L2X model[3] most useful as it is more efficient than LIME and easier to use than the anchors introduced by [60].

Based on the results of the algorithm simple hypothesis can be derived and adversarial attacks can be generated. To ensure diversity in adversarial attacks the manual creation of adversarial samples is most appropriate and realistic to challenge the model. These identified algorithms will be used in the further course of this work.

[3]https://github.com/Jianbo-Lab/L2X

Chapter 4

Methodology

This chapter presents the detailed methodology which is used to answer the research questions and follows the research methodology in section 1.4. The structure of this chapter is based on the mentioned 4 pillars and the last two process steps (1) design and optimization of the model; (2) demonstration and evaluation. Each pillar represents an objective and answers one of the four main research questions. A diagram at the beginning of each section illustrates the detailed stages of this particular part of the methodology.

4.1 Multi-task Training

The following section is structured in three parts. Within the first part, we describe the selection process that we applied to determine the different downstream tasks that were used in the multi-task pre-training. In the second part we explain the used model architectures and the multi-task pre-training process. Lastly, we elaborate the fine-tuning process of the pre-trained T5 model. This section describes the design and optimization of the multi-task training approach using the selected SciEntsBank dataset. The overall strategy can be seen in figure 4.1.

4.1.1 Task Selection for Multi-task Training

In this section we describe the details about the methodology for choosing the datasets for multi-task training and introduce the datasets.

Since the overall goal of the published T5 model was to study the general learning abilities the original model was trained with 23 datasets. This is connected with a huge computational effort due to the model architecture and its size. In order to improve the efficiency of training, we tried to limit the number of used datasets by profoundly pre-selecting only related datasets that promise potential added value. As already described in the related work section, this step is based on the assumption that a model learns relevant (task-related) features instead of general features and thus increases the performance for ASAG. Simultaneously, with this approach we guarantee a more efficient training process since we limit the number of training tasks.

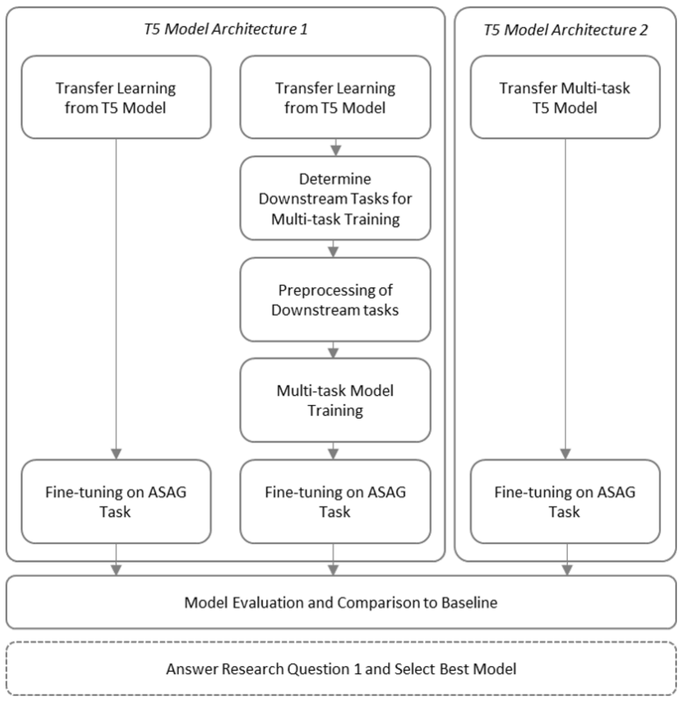

T5 Model Architecture 1
T5 Model Architecture 2
Transfer Learning from T5 Model
Transfer Learning from T5 Model
Transfer Multi-task T5 Model
Determine Downstream Tasks for Multi-task Training
Preprocessing of Downstream tasks
Multi-task Model Training
Fine-tuning on ASAG Task
Fine-tuning on ASAG Task
Fine-tuning on ASAG Task
Model Evaluation and Comparison to Baseline
Answer Research Question 1 and Select Best Model

Figure 4.1: Methodology for evaluating multi-task training approaches

Therefore, we analyzed the existing datasets in the NLP field and applied a filter strategy which is illustrated in figure 4.2 to identify the most suitable datasets.

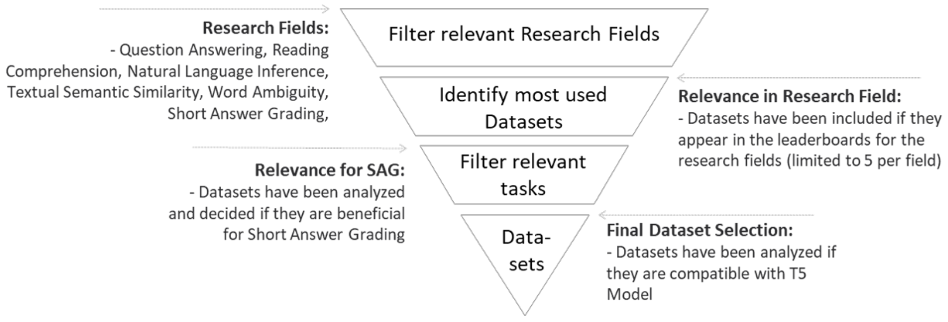

Research Fields:
- Question Answering, Reading Comprehension, Natural Language Inference, Textual Semantic Similarity, Word Ambiguity, Short Answer Grading,
Filter relevant Research Fields
Identify most used Datasets
Relevance in Research Field:
- Datasets have been included if they appear in the leaderboards for the research fields (limited to 5 per field)
Relevance for SAG:
- Datasets have been analyzed and decided if they are beneficial for Short Answer Grading
Filter relevant tasks
Data-sets
Final Dataset Selection:
- Datasets have been analyzed if they are compatible with T5 Model

Figure 4.2: Filter strategy to identify suitable multi-task datasets

In the first step we defined five similar research areas that are related to ASAG that are particularly promising. These are listed below with the respective assumptions regarding their benefits for ASAG:

1. Natural Language Inference: By including tasks from NLI we expect the model to increase its performance on ASAG by learning to predict the semantic relations between sentences. This corresponds to the semantic relation between reference and student answer.

2. Semantic Textual Similarity: The goal is that the model learns to detect sentence paraphrases and therefore increases the ability to capture semantic information in the student answers and apply this knowledge to the corresponding reference answer.

3. Question Answering and Reading Comprehension: We expect the model to learn commonsense reasoning and answering questions especially with regard to additionally provided context information. By gaining such knowledge the authors expect the model to apply this reasoning and answering behavior to the ASAG task.

4. Word Ambiguity: Since these tasks deal with an important and difficult problem in NLP we expect that the model further learns to understand references between objects/subjects and the corresponding pronoun as well as understands the meaning of words in a different context.

5. Short Answer Grading: Due to the similar nature of the task, the author expects that the models learns about the syntax of student answers and in particular the nature of question, reference answer and student answer.

In each of these areas we identified the most popular datasets. These are illustrated in table 4.1.

Table 4.1: Original selection of datasets for each research field

Research Field	Datasets
Natural Language Inference	[74] [75] [76] [77] [78]
Semantic Textual Similarity	[79] [80] [81] [82]
Question Answering and Reading Comprehension	[83] [84] [85] [86] [87] [88]
Word Ambiguity	[89] [90] [91]
Short Answer Grading	[32] [33] [31]

This selection was further analyzed in greater detail and decided whether the dataset is compatible with the T5 model. After the selection process, we were left with a total of 14 datasets. Table 4.2 gives an overview of the selected datasets, their size[1] and corresponding research field. In order to use the datasets with the T5 model architecture, we first had to transform each into a text-to-text format. Detailed examples of the transformed model inputs for each task are illustrated in the appendix A.1. These were partly based on the transformations applied in [3] or individual formats were defined.

4.1.2 Multi-task Training and Fine-tuning

The goal of this first stage of the methodology is threefold. (1) we investigated the usefulness and superiority of a multi-task pre-training approach; (2) we determined if a profound dataset selection strategy can increase the performance; (3) we aimed to outperform the state-of-the-art model since we want to obtain a high-performance model.

In order to achieve these goals, firstly a suitable model had to be chosen and an adequate methodology needed to be defined. As mentioned in the related work section we used the T5 model architecture provided by the research team from Google [3]. The model is available

[1]Only the training data was used in order to keep the option of a model validation and test on other tasks

Table 4.2: Overview of the selected multi-task datasets

Nr.	Dataset	Research Field	Train Sample
1	MultiRC: Multi-Sentence Reading Comprehension [87]	Question Answering	27,243
2	SQuAD 1.0: The Stanford Question Answering Dataset [84]	Question Answering	87,599
3	ReCoRD: Reading Comprehension with Commonsense Reasoning [86]	Reading Comprehension	60,621
4	SNLI: Stanford Natural Language Inference (SNLI) Corpus [74]	Natural Language Inference	549,367
5	MulitNLI: Multi-Genre Natural Language Inference (MultiNLI) Corpus [75]	Natural Language Inference	391,165
6	RTE: Recognizing Textual Entailment [78]	Natural Language Inference	2,490
7	COPA: Choice of Plausible Alternatives [77]	Natural Language Inference	400
8	STS Benschmark [80]	Semantic Textual Similarity	5,749
9	MRPC: Microsoft Research Paraphrase Corpus [81]	Semantic Textual Similarity	4,439
10	SICK Dataset [79]	Semantic Textual Similarity	4,439
11	WiC: The Word in Context Dataset [91]	Word Ambiguity	5,428
12	WSC: The Winograd Schema Challenge [90]	Word Ambiguity	554
13	ASAP-SAS Dataset [33]	Short Answer Grading	17,207
14	University of North Texas Dataset [32]	Short Answer Grading	2,442

in several sizes as explained in section 2.3. Due to the associated computational costs, we used the second largest 3B model with roughly 3 billion parameters rather than the largest with 11 billion. The underlying model code from the authors' and their model checkpoints are provided in their GitHub repository [2].

Since the authors' only released certain model checkpoints for use, two different model architectures had to be used for the experiments. This made a reliable comparison with the published model impossible. Therefore as illustrated in figure 4.1, we used the same model checkpoint for both experiments in section 4.1.2 but a different model architecture for the fine-tuning of the already pre-trained model in section 4.1.3. As a consequence, these should not be compared against each other since they vary in their parameter choice and architectural setup. The architectural differences between the two models can be seen in table 4.3.

[2]https://github.com/google-research/text-to-text-transfer-transformer

Multi-task Training Experiments

With the first two conducted experiments we aimed to achieve the first two goals mentioned in section 4.1.2. To determine the base performance of the model we conducted the first experiment where we used the released checkpoint[3] from the t5.1.1.xl model (corresponds to the T5 Model Architecture 1 in table 4.3) and fine-tuned it on the target dataset (subsection 5.1) without additional multi-task pre-training.

For the comparison and to decide if the multi-task training is beneficial we conducted the second experiment. We loaded the same t5.1.1.xl model from the released checkpoint and further trained it on the multiple tasks selected in section 4.1.1. Afterward, we fine-tuned the model on the target dataset and reported the performance. Finally, we compared the performance on the test set according to the evaluation metrics introduced in section 2.4, and concluded to what extent the used multi-task training approach was beneficial.

4.1.3 Fine-tuning of Pre-trained Model

In the third experiment, we wanted to improve the performance further by using the extensive pre-trained model checkpoint from the 3B model and fine-tune it on the target task. This model refers to the Architecture 2 in table 4.3. The model performance is again evaluated on the test set and compared against the defined baseline in section 3.1.2. To determine the model with the highest performance, the three applied models were compared and the best was the basis for further experiments.

Table 4.3: Architectural differences and key parameters of used models

Category	T5 Model Architecture 1	T5 Model Architecture 2
Activation function	GEGLU	ReLU
Dropout Rate	0.1	0.1
Pre-trained	C4 corpus	C4 corpus and mulit-task
Parameter Sharing[4]	No	Yes
d_{ff}	5120	16384
d_{kv}	64	128
d_{model}	2048	1024
Attention Heads	32	32
Number of Layers	24	24

4.2 Domain Adaptation with Domain-specific Fine-tuning

This section of the methodology focuses on answering the second research question. In detail, we aim to investigate if we can improve the fine-tuning process by means of domain

[3]https://github.com/google-research/text-to-text-transfer-transformer/blob/master/released_checkpoints.md

adaptation. This answers the question to what extent a data sparse, domain-specific fine-tune process or a non-domain-specific fine-tune process is superior. The detailed methodology is illustrated in figure 4.3

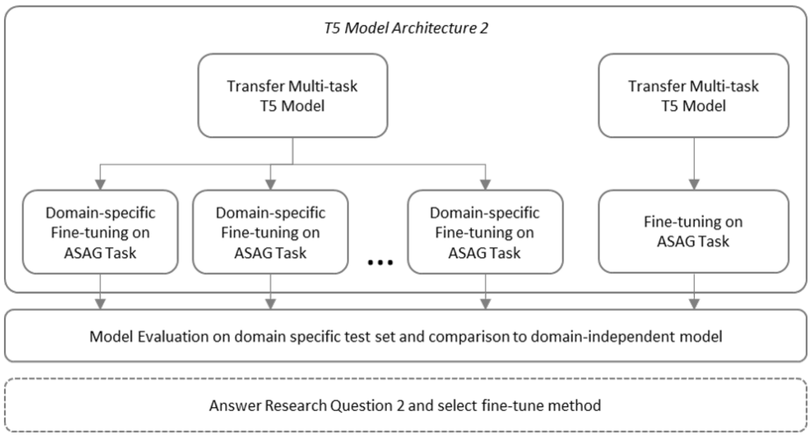

T5 Model Architecture 2
Transfer Multi-task T5 Model
Transfer Multi-task T5 Model
Domain-specific Fine-tuning on ASAG Task
Domain-specific Fine-tuning on ASAG Task
...
Domain-specific Fine-tuning on ASAG Task
Fine-tuning on ASAG Task
Model Evaluation on domain specific test set and comparison to domain-independent model
Answer Research Question 2 and select fine-tune method

Figure 4.3: Methodology for multi-task domain-specific fine-tuning

In order to analyze the performance of a domain-specific model we fine-tuned one model for each domain separately. This means that each model is fine-tuned only on the relevant data from the corresponding domain. For that reason, we split the train, validation and test set into 12 parts where each part contains question-answer pairs from their particular domain. Since the datasets are relatively small, we trained three different models per domain. Each of them is trained on a different training/validation split. By evaluating the average test score we make sure that the results are reliable despite the small data size. The final model evaluation is done by comparing the performance on each domain with the performance of the previously selected model. Therefore, we used the already fine-tuned model and analyzed the performance on the test set for each domain. For evaluation and comparison we used the weighted average f1-score.

4.3 Model Explainability and Interpretability

As third part, we elaborate the methodology regarding the model explainability and interpretability. This is the first step in the demonstration and evaluation of the determined training and fine-tune method. In detail, we describe the three different stages of the model analysis process. For each stage, we explain the chosen strategy and model choices that we used in order to interpret and analyze the model behavior.

4.3.1 Introduction and Overview of Methodology

This section addresses the problem that the predictions of a practical ASAG model need to be comprehensible. Hence, we aim to gain a general understanding of the basis of model decisions. The main focus is to determine why and when the model predicts a full score and if these predictions follow certain answer patterns. The underlying analysis strategy is illustrated in figure 4.4

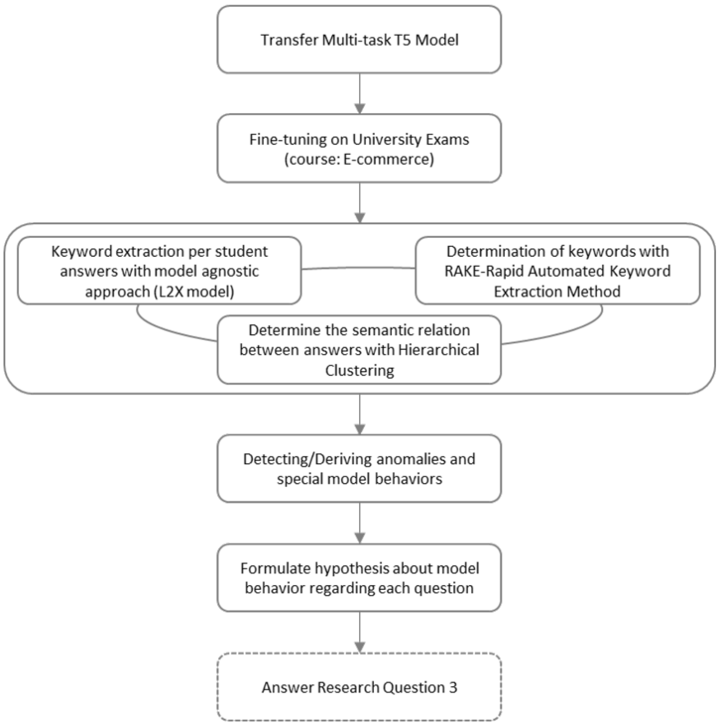

Transfer Multi-task T5 Model
Fine-tuning on University Exams (course: E-commerce)
Keyword extraction per student answers with model agnostic approach (L2X model)
Determination of keywords with RAKE-Rapid Automated Keyword Extraction Method
Determine the semantic relation between answers with Hierarchical Clustering
Detecting/Deriving anomalies and special model behaviors
Formulate hypothesis about model behavior regarding each question
Answer Research Question 3

Figure 4.4: Methodology for model explainability and interpretability

As figure 4.4 shows, we first used the optimized model and fine-tuned it on the extracted dataset from the University of Twente. Thereby we wanted to demonstrate to what extent the model is applicable to a real university context. Detailed dataset description and the data pre-processing are illustrated in section 5.2. After fine-tuning the model, the respective predictions from the training, validation and test set were extracted. To reduce the analysis effort we selected five suitable questions for a deeper analysis. We then carried out an analysis process structured in three stages: (1) Analyzing false positive answers, (2) extracting keywords and identifying local key features and (3) analyzing the semantic relations between the answers. Based on the combined results we formulated hypotheses that describe the model predictions.

4.3.2 First Stage: Analyzing False Positives

In the first stage we analyzed the false positive (predicted value was 2 but the actual value was less) cases. This analysis was done by manually skimming through the answers in order to identify patterns. In particular we paid attention to similar semantic information and overlapping phrases within student answers. The goal of the analysis was to identify commonalities within the false positive predictions in order to derive model behavior.

4.3.3 Second Stage: Extracting Keywords and Local Key Features

In the second stage, we analyzed the model behavior in greater detail by means of model-agnostic approaches explained in 3.4.2. The central goal was to determine the keywords and key features for each question and predicted labels in order to derive the important keyphrases for the predictions. For this, we used two algorithms to consider both global and local features.

First, we analyzed global features for particular questions. Therefore, we determined keywords-phrases for each question and answer category (i.e. predicted score) by means of the RAKE (Rapid Automatic Keyword Extraction) method [92]. This is a method that focuses on identifying keyphrases with frequently used content words. We used this method to imitate the grading process of teachers that search for particular keyphrases within an answer. Such recurring response patterns can be an indication of a certain prediction behavior. We extracted the main keywords based on the ratio of the word degree and frequency since we are more interested in context-relevant keyphrases. However, this implies that single keywords will get a lower score. Therefore, we extracted all the keyphrases with their corresponding scores and investigate the top 10 for each keyphrase (unigrams, bigrams and trigrams).

Second, we applied the L2X model to identify local features from the individual answers (important keyword unigrams). We did this by using the L2X model implementation since it focuses on local keywords per answer rather than for the entire question/score. This method was chosen because of the versatile nature of student answers. The goal was to extract local answer features for a single answer. The underlying idea was to identify a specific number of keywords for every single answer. These keywords were then grouped per question and category and overlaps (i.e. commonalities) were identified. These local features (i.e. uni-gram keywords) also provide valuable insight later in the manual analysis of semantic relations between answers.

4.3.4 Third Stage: Identifying Semantic Relations

In the third stage, we aimed to identify the semantic relations within groups of answers. The first goal is to determine how semantically related the answers are and if semantic patterns were observable. In detail, we identified different topic areas - based on the semantic relations between answers - that get high scores. This was done because the dataset encompasses open questions on several topic areas. Within questions, we assumed that students mention thematically related answers. The second goal was to identify certain anomalies within semantically related student answers. This was done simultaneously in a manual manner. In order to simplify the manual analysis process we used a hierarchical cluster algorithm which groups semantically similar answers. We chose this algorithm because we can first look at the corresponding semantic variances of the answers (by visualizing dendogram) and determine a preferable number of clusters. With this we ensured that significantly different answers are not clustered together.

4.3.5 Putting it Together: Formulating Model Hypotheses

Finally, we brought these three analysis stages together and derived specific anomalies or prediction characteristics. Based on this, we determined hypotheses for each question which reflect and explain why a model predicts an answer as correct.

4.4 Model Robustness against Adversarial Attacks

This section contains the chosen methodology to evaluate the robustness of the demonstrated model. We did this by generating adversarial attacks based on the identified model behavior. The detailed steps are illustrated in figure 4.5. We incorporated two different goals with the used methodology.

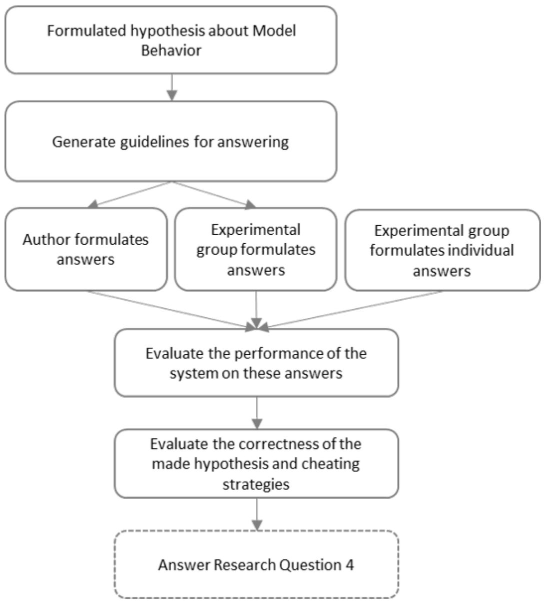

Formulated hypothesis about Model Behavior
Generate guidelines for answering
Author formulates answers
Experimental group formulates answers
Experimental group formulates individual answers
Evaluate the performance of the system on these answers
Evaluate the correctness of the made hypothesis and cheating strategies
Answer Research Question 4

Figure 4.5: Methodology for evaluating model robustness

The first goal was to evaluate the accuracy of the generated hypotheses from the previous analysis. We took these and generated guidelines that reflect how model features can be exploited. This included the instruction that the keywords identified in section 6.3 should appear in the answer and that the answer should not make sense or should be unrelated. The central goal was to evaluate the applicability of the identified hypotheses. Within this method, an experimental group of students was asked to find answers in order to trick the model into predicting an originally wrong answer as correct. We further supplemented the student answers by creating more answers. These answers follow similar guidelines but are expanded according to the knowledge gained from previous sections. In this way, we wanted to focus even more specifically on the supposed weak points of the model.

For the second goal, we aimed to determine the capability of the model to deal with student manipulations by challenging the model in a more general way. In detail, we generated several wrong answers based on specific cheating strategies. Wrong in this context includes syntactically wrong sentences, off-topic answers, meaningless answers and more. A detailed description of the cheating strategies can be seen in the appendix table A.4. This analysis is essential to determine the usability of the model for a real-world application. In which such an ASAG model needs to be able to handle students whose goal is to muddle through questions and collect points with their answers.

Chapter 5

Datasets

For the conducted research several datasets are of particular relevance. In this chapter we describe the main datasets used for evaluating the performance and their characteristics. Followed by the introduction and the undertaking pre-processing steps of the second dataset which is used to demonstrate and evaluate the model in a university context.

5.1 SciEntsBank Dataset

The SciEntsBank dataset is part of the task introduced in [31] and became well known as part of the the SemEval 2013 Shared Task 2013 challenge. The SciEntsBank data contains assessment questions and student answers from students in grades 3-6 in schools across North America.

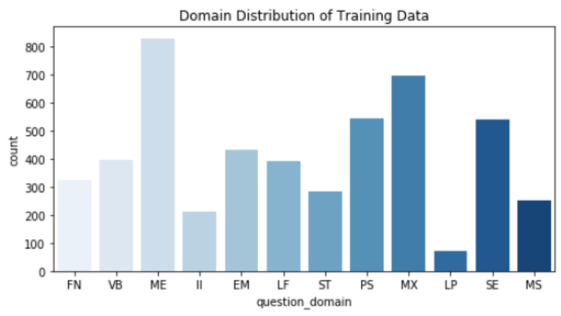
In general, the goal of the task is to identify correct answers, common mistakes like omissions and wrong or thematically irrelevant statements for developing customized correction strategies. The dataset is available in three different versions but does not differ in content, since these versions use the exact same data and differ only in the used labels. For the research, we focused on the 5-way classification task since multi-class classification is more challenging. As the name suggested the 5-way task consists of five labels. Where each question-answer-pair is labeled as either (0) contradictory ; (1) correct; (2) partially correct incomplete; (3) non domain or (4) irrelevant/incorrect. The 5-way task was especially created for the tutoring dialogue system since the labels indicate already a possible dialogue-feedback strategy.

In total the dataset contains 4,969 answers to 135 different questions from 12 domains. Detailed dataset properties can be taken from table 5.1. The dataset has an uneven domain distribution of samples. This can be seen in figure 5.1 a). It can be seen that few domains (e.g. ME, MX, PS, SE) make up a large part of the total training data.

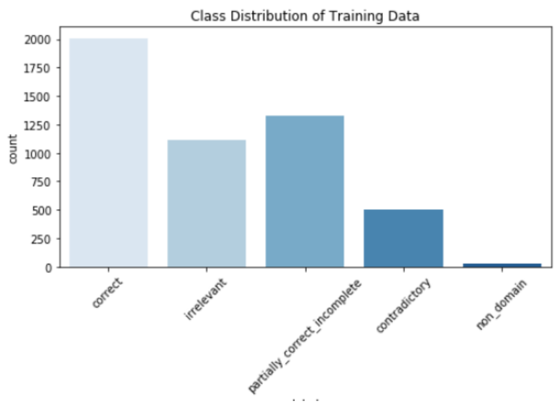
In addition, as picture 5.1 b) shows, the class distribution of the SciEntsBank dataset is imbalanced and skewed towards correct and partially correct answers. This property can be problematic for machine learning algorithms since they are mostly designed under the assumption of even class distribution. Despite the risk that this imbalance can lead to a poor performance of the minority class we decided to accept this imbalance and not to use

Table 5.1: Overview of the dataset and key properties

Category	Train Dataset	Test Dataset
Number of samples	4,969	5,835
Number of Domains	12	15
Number of Questions	135	196
Average Number of Answers per Question	37	30
Average Student Answer Length (in words)	13	11
Average Reference Answer Length (in words)	17	15
Label Range	{0,1,2,3,4}	{0,1,2,3,4}



(a) Domain distribution of the training Data



(b) Class distribution of the training Data

Figure 5.1: Properties of the SciEntsBank dataset

re-sampling methods such as [93] or [94]. The decision to excluding this from the scope of the work was made because it is a complex problem and a comprehensible analysis was not feasible in the given time.

In order to test the performance of the model, the authors' of the dataset introduced a test set that consists of three different categories where each is capturing a different model behavior. First part is the testing of the system regarding unseen answers. The second part deals with unseen questions on which the model was not trained on. The last part of the testing set deals with different questions from unseen domains. With these three categories, the dataset captures the flexibility of the model and capability to generalize over different domains.

5.2 University of Twente Dataset

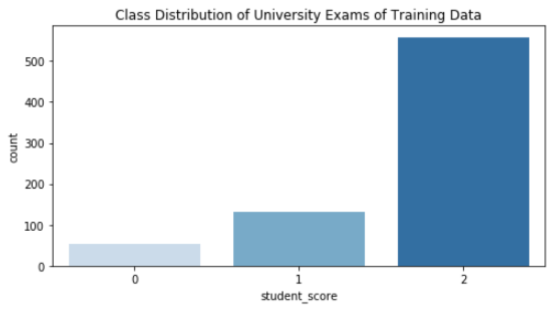
In order to analyze the performance of the identified model on a real-world example we used a dataset obtained from the University of Twente. The dataset consists of exam questions from the master course Electronic Commerce which was taught in 2019/2020 (course: 192320501). With the help of the professor of the course the dataset was extracted from the digital assessments tool Remindo and several pre-processing steps have been applied.

After pre-processing the dataset consisted of 1243 samples distributed over 22 different questions which were structured in 8 main question with several sub-questions. The detailed properties of the dataset can be seen in table 5.2.

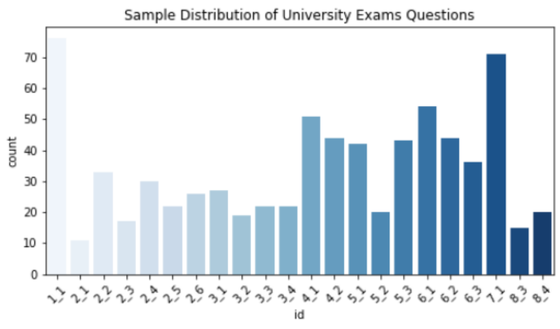
Table 5.2: Key properties of the university dataset

Category	Train Dataset	Test Dataset
Number of samples	994	249
Number of Questions	22	22
Average Number of Answers per Question	45	11
Average Student Answer Length (in words)	35	33
Average Reference Answer Length (in words)	31	32
Label Range	{0,1,2}	{0,1,2}

A general analysis of the dataset distribution shows significant class imbalances. As figure 5.2 a) illustrates, the majority of the samples are allocated in class two which represents the class of correct answers. Furthermore, the sample distribution over the different questions is imbalanced which can be seen in figure 5.2 b). It further shows that only two questions encompass around 70 answers whereas the majority of the questions contain less than 40 question-answer pairs.



(a) Class distribution of training data



(b) Sample distribution of questions

Figure 5.2: Properties of the university dataset

5.2.1 Data Pre-processing

In this section we describe the pre-processing steps that were used to restructure the exam data and adapt the input format for the transformer model.

Question Pre-processing and Reformulation

Since the questions and answers of the dataset were of good overall structure it was possible to restructure them by combining the answers from questions. For instance, one question was to name three different advantages where the original answers were split into three different answers and were graded separately. Thus, it was possible to treat each of the three answers from one student as a single answer by simply reformulating the question. Other questions expected the student to comment on a statement by either agreeing or disagreeing, providing two reasons in each case. These two reasons were also graded separately by the professors. This allowed us to split the answers and assign two different questions by transforming one question to two and only consider the relevant answers. To illustrate the procedure one of the original questions can be seen in the first text box whereas the reformulated questions in the second box.

Below you can find three propositions relating to E-commerce. First, specify whether you agree or disagree with the propositions. Then, provide TWO reasons for your choice (each is two points). Give key facts from the course materials that support your view.
1. B2C commerce in Europe is now the dominant way of shopping across all retail sectors.

The reformulated questions are the following:

1. Do you agree with the fact that B2C commerce in Europe is now the dominant way of shopping across all retail sectors? (only answer if you agree)
2. Do you disagree with the fact that B2C commerce in Europe is now the dominant way of shopping across all retail sectors? (only answer if you disagree)

By choosing these pre-processing approaches it was possible to amplify the number of training samples for several questions. Furthermore, the question and answer structure better fit the model due to the division of the questions into answers that agree and disagree. Hence, the model learns the distinctions better since the argumentation is not of opposite nature. Similar steps were done for several questions. After determining all the questions we adapted the reference answers provided by the professors and created new ones that match with the transformed questions.

Pre-processing of Student Answers

Since the model itself used a specific tokenizer and pre-processing, we only discuss the specific pre-processing steps to clean the data. Firstly, we replaced all line or paragraph breaks (e.g. '\n, \n\n') with a single space. Secondly, since students tend to include enumerations in their answers combined with hyphens, the author replaced '\n-' with a dot. Lastly, we deleted special characters (only cases with several dots, e.g. '...') and several minor changes in terms of white space characters (e.g. " ." to "."). In addition, some of the answers exceeded the maximum answer length of 100 and were therefore excluded since these are not considered short answers according to [5].

Label Pre-processing

The original prediction task was a regression task since the model answers are theoretically continuous. However, most of the professors evaluate the exams according to certain points which results in a limited amount of possible scores. Therefore, the task can be interpreted as a classification task due to its deterministic labels. Since the dataset is so small we decided to pre-process the labels in order to create a dataset with a more-utilizable uniform class distributions. Most of the label ranges of the questions were 0,1,2, the labels in between (e.g. 0.5, 1.5) were rounded off and assigned the corresponding class. This approach was chosen for all questions and their sub-questions.

Chapter 6

Experiments and Results

In the current chapter the experimental setup will be elaborated. We will include an in-depth description of the specific implementations and the obtained results from the different parts of experiments as described in chapter 4. This is followed by a short interpretation.

6.1 Multi-task Training

The goals in this section were to conclude if the multi-task training approach is beneficial for ASAG and if the baseline (defined in section 3.1.2) can be outperformed. For this reason we conducted three experiments in which we implemented and tested different models. Each model implementations is based on their released code on github[1]. For the evaluation of the trained models we compared them against each other and the defined baseline.

6.1.1 Experiment 1: Fine-tuning without Multi-task Training

In the first experiment we determine the base performance of the model that was not trained in a multi-task manner. For this we used the pre-trained model checkpoint which was only trained on the C4 corpus and not on multiple tasks.

The model refers to the implementation of the T5 Model Architecture 1 in table 4.3 where we used the released checkpoint from the t5.1.1.xl model. As parameter setting we used a learning rate of 0.001 since this was proposed by [3] as a result of a comprehensive study on the hyper-parameter choices for the model. Furthermore, we used a relatively small batch size of 16 since we wanted to decrease the computational costs and a large batch size would further increase them [95]. Although preferable, a too large batch size would have caused problems, because we were already working with the model on the edge of the memory capabilities. We fine-tuned the model for 25,000 steps without early stopping criteria and selected the model with the highest validation score. Furthermore, we determined the length of the input sequence as 512 and the output sequence as 5. The reason for this is that for a smaller output sequence the model predicts empty string. However, this should be as low as possible since we only want to predict the target labels.

[1]https://github.com/google-research/text-to-text-transfer-transformer

These parameters were set as default and were not altered since we did not conduct a hyper-parameter analysis due to time constraints and disproportionate computational effort. Unfortunately, the implementation does not allow the training score to be extracted to check if the model is over- or underfitting. Therefore we only compared the validation and the test score. The results for this experiment corresponds to the acronym *T5-XL-base* in table 6.1. Furthermore, the test score in the table corresponds only to the category *Unseen Answer* since otherwise it would have diluted the results and the informative value.

Table 6.1: Weighted average f1-score for validation and test data for each experiment

Exp.	Model	Validation Score	Test Score
1	T5-XL-base	0.6036	0.5317
2	T5-XL-mt	0.6483	0.6096
3	T5-3B-base	0.7765	0.7451

It can be observed that the model achieved an overall test score of 0.5317 which is a below average result. Despite the difference between validation and test score, it seems that the model does not over-fit the data considering the difficulty of the task. The detailed results for each category can be seen in table 6.2. In each of the categories the *T5-XL-base* model is inferior to the baseline. However, this was to be expected since this model was not trained extensively and it only served as a basis for comparison.

6.1.2 Experiment 2: Multi-task Training and Fine-tuning

After the baseline has been determined for the model we applied the individual multi-task training approach and further pre-train the model. This was done since we want to identify if the performance can be improved with a more profound and efficient pre-training approach.

In this experiment we used the same multi-task training approach setup as the authors of [3]. We first prepared the different tasks by generating the corresponding pre-processing functions and by combining the different tasks. For each task we used the weighted average f1- score as optimization metrics and cross-entropy loss. The transformed input formats for each task are illustrated in the appendix section A.1. In the training process, randomly selected samples from each task are assigned to a batch and the model parameters are adjusted accordingly. This was done by using the examples-proportional mixing approach without setting a dataset limit (for details see section 2.3.5). Therefore, we considered each task according to the overall dataset distribution.

For multi-task training we used the same checkpoint as in section 6.1.1 with the same parameter settings. As training step size we used $2^{19} = 524,288$ since it was used by [3] for pre-training. With this step, we tried to imitate the training method (i.e. the number of training steps) used in the paper. For the training we included training and validation samples of the corresponding tasks. However, due to the technical structure of the implementation it was not possible to extract the validation scores for each specific task. Hence, we only saved few checkpoints of the model during the training. For this reason, the last checkpoint of the model is used for the following fine-tuning. By doing so we did not consider to what extent the model might be over-trained on specific tasks. After the pre-training of the model we fine-tuned it with the same hyper-parameter choices as in the previous section 6.1.1. After fine-tuning we used the model with the best validation score and applied it on the test set.

The results of the fine-tuned model *T5-XL-mt* can be seen in table 6.1. The model achieved a validation score of 0.6483 and a test score of 0.6096 which indicates that the model does not over or under-fit the data in a significant manner. A more detailed analysis of the model performance by category is shown in table 6.2. It can be seen that the training was successful with regard to the category Unseen Answer, as the weighted average f1-score was improved by more than 0.07. However, this was at the expense of the other two categories, which have deteriorated significantly. This score of less than 0.3 in each category is slightly better but almost equivalent to randomly chosen predictions. This led to the average of the model slipping below 0.4 which is worse than the average result of the previous model. Despite the positive result for unseen answers, the model was not able to beat the baseline where it performed much worse than previous models.

Furthermore, the results indicate that the training was positive for the category unseen answer. Here the model has learned to make better predictions when it is trained on our task selection. However, the generalization ability of the model suffered since the performance for unknown questions and domains worsened significantly. Therefore, it seems that the chosen multi-task training approach is only beneficial for questions the model has already been seen during training. The reasons behind this behavior are not clear but it seems that the model learned more during training and fine-tuning while simultaneously losing its ability to generalize over unseen domains and questions.

6.1.3 Experiment 3: Fine-Tuning with Pre-trained T5 Model

The models from previous sections were not able to achieve new state-of-the-art results on the target dataset. Therefore, we used the originally published model which has been trained much more extensively in order to improve the performance against the baseline. The model was used identically to the previous ones by accessing the checkpoint of the model. As explained in the methodology (section 4.1), this is the same type of model with a different model architecture, configuration choices and pre-training method.

We implemented the model based on the released checkpoint described in section 4.1.3. All parameters for fine-tuning were kept the same.

The model *T5-3B-base* achieved a test score of 0.7451 with a validation score of 0.7765. This indicates that the model learned the question-answer-structure from the training and that there was no indication of under or over-fitting. The detailed performance of the model *T5-3B-base* can be seen in the table 6.2. The model outperformed the baseline and achieved new state-of-the-art results on the target dataset. In the category unseen answer, the model is approximately 0.07 better than the baseline. For the other two categories, the model achieved a 0.10 (for unseen domain) and 0.15 (unseen question) higher weighted average f1-score than the baseline. Interestingly, the model achieved the same performance for unknown questions and unknown domains as the baseline for unknown answers. This indicates how powerful such exhaustively trained and complex models are and how well they represent textual data.

6.1.4 Evaluation of Multi-task Training Experiments

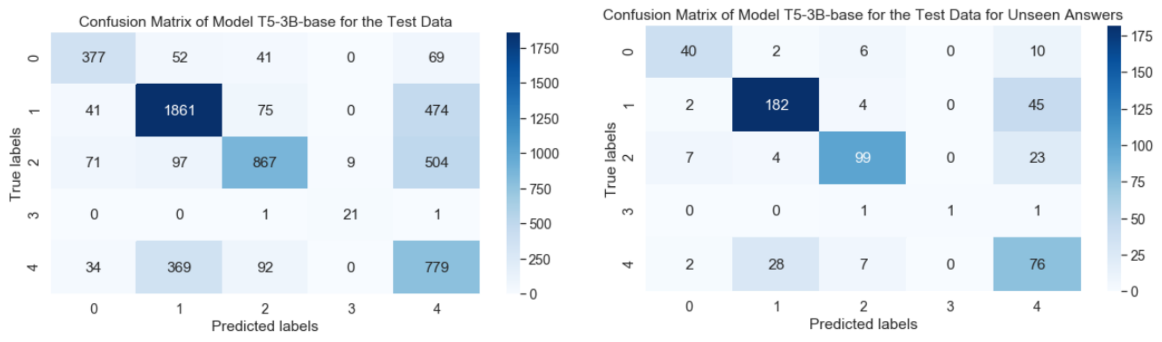
The following table 6.2 illustrates the detailed results on the SciEntsBank test set achieved by each of the three models.

Table 6.2: Weighted average f1-scores of conducted experiments and baseline

Exp.	Model	Average**	Unseen Answer*	Unseen Question*	Unseen Domain*
1	T5-XL-base	0.4906	0.5317	0.4496	0.4121
2	T5-XL-mt	0.3875	0.6096	0.2676	0.2855
3	**T5-3B-base**	**0.6975**	**0.7451**	**0.6687**	**0.6788**
	Baseline	0.6565	0.6720	0.5310	0.5740

The results show that the excessive pre-trained *T5-3B-base* model outperformed each of the individually trained models *T5-Xl-base* and *T5-Xl-mt* as well as the baseline in each category significantly. This makes it the new-state-of-the art model for the SciEntsBank dataset.

We analyzed the best model in greater detail by means of the confusion matrix split into figure 6.1a for all three categories and figure 6.1b for the single category unseen answers[2].



(a) Confusion matrix for all categories

(b) Confusion matrix for unseen answers

Figure 6.1: Confusion matrix of the T5-3B-base model on test set

The two confusion matrices make it clear that the general prediction behavior within the different classes did not differ much between the categories. Due to these minor differences, the following performance metrics refer only to the category unseen answers (figure 6.1 b) but are also representative for the overall model. In detail it shows that the model is able to predict the correct answers well considering the difficulty with a precision of 0.8425 and

[2]Label decoded: 0 contradictory, 1 correct, 2 irrelevant, 3 non domain, 4 partially correct

a recall of 0.7811. However, for the partially correct the model only achieved a precision of 0.4903 and a recall of 0.6725. Many partially correct answers are classified as correct which shows that the model lacks in distinguishing between correct and partially correct answers and has a general bias towards correct answers. In addition, the model falsely predicts contradictory, irrelevant and correct answers as partially true. This is the main reason for the low performance for the class of partially true answers. Although the model in general performs well, the results indicate that the model has difficulties especially in classifying partially true answers.

6.2 Domain-specific Multi-task Training

In this section, we describe the implementation of the fourth experiment. In detail, we elaborate the preparation of the domain-specific data, the experimental setup for the fine-tuning process and report the corresponding results for each domain. The goal in this experiment was to investigate to what extent a domain-specific fine-tuned model can improve the overall performance and if it helps to mitigate the data sparsity problem. The conducted experiment is based on the best performing model architecture *T5-3B-base* from the previous section.

6.2.1 Experiment 4: Domain-specific Fine-tuning

Since the previously implemented model has been trained on the entire dataset (i.e. all 12 domains) we first split the train, validation and test set according to their domains. This resulted in 12 different train/validation/test sets which served as input for the domain-specific fine-tuning. This step made the already small training samples even smaller. This caused problems since fine-tuning models on such a small dataset is likely to lead the network to overfitt the data. Consequently, the reliability of the training suffers. In order to avoid reduced reliability of the performance and to maintain the informative value of the approach, we created for each domain 3 different train/validation splits. By that, we trained three different models for each domain and averaged their performance on the test set for each domain. In addition, we calculated the standard deviation for each result to measure the variation within the splits. This increased the significance of the chosen approach and compensated partially for the small dataset. Each domain is represented in different quantities in the data and we wanted to train each for the same number of epochs. Therefore, we adapt the step size for each domain accordingly and reduced the batch size. The detailed parameters for each domain-specific model training can be seen in the appendix table A.1. All the other previously used fine-tuning hyper-parameters are kept the same. After the models have been trained we analyzed the prediction of the previously implemented model and reported the performance on the test set for each domain independently.

In table 6.3 the results of the domain-specific and non-domain-specific model performances are illustrated. The latter corresponds to the *T5-3B-base* model from the previous

Table 6.3: Results for domain-specific fine-tuning per domain

	EM		**FN**		**II**		**LF**	
Category	UA/σ	UQ/σ	UA/σ	UQ/σ	UA/σ	UQ/σ	UA/σ	UQ/σ
Domain	0.678/0.048	0.408/0.004	0.619/0.004	0.268/0.101	0.619/0.006	0.533/0.014	0.571/0.054	0.465/0.064
Non-Domain	0.817	0.503	0.777	0.691	0.649	0.747	0.673	0.528
Test Samples	48	80	36	40	24	38	44	40
	LP		**ME**		**MS**		**MX**	
Category	UA/σ	UQ/σ	UA/σ	UQ/σ	UA/σ	UQ/σ	UA/σ	UQ/σ
Domain	0.791/0.071	0.266/0.011	0.756/0.022	0.447/0.032	0.673/0.043	0.421/0.036	0.605/0.029	0.821/0.089
Non-Domain	0.881	0.671	0.769	0.641	0.791	0.533	0.672	0.933
Test Samples	8	40	92	80	28	40	80	40
	PS		**SE**		**ST**		**VB**	
Category	UA/σ	UQ/σ	UA/σ	UQ/σ	UA/σ	UQ/σ	UA/σ	UQ/σ
Domain	0.834/0.034	0.683/0.088	0.612/0.026	0.372/0.045	0.641/0.049	-	0.547/0.02	0.547/0.021
Non-Domain	0.858	0.773	0.741	0.667	0.708	-	0.663	0.624
Test Samples	44	215	60	80	32	-	44	40

experiment where we reported the performance for each domain. For each model and domain the results are reported separately for each category (unseen answer, unseen question). The results for the domain-specific model in table 6.3 are averages of the three separately trained models. The detailed results for each split are illustrated in table A.3 in the appendix. It becomes clear that the model which has been trained on all domains achieved better performance on each domain and demonstrated its superiority. For some domains the standard deviation of the splits is high, which indicate the issue with the small training set and that some of the models overfit the data. This pattern can be seen in the detailed results when we take the corresponding validation score into account.

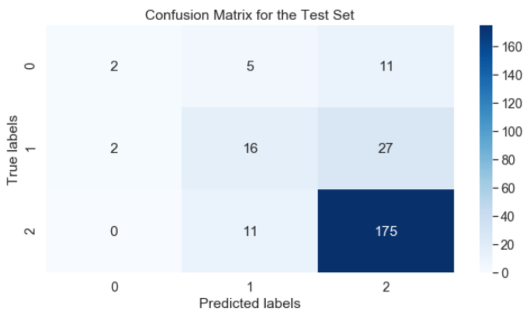
6.3 Model Explainability and Interpretability

Within this section, we describe the implementation of the identified model and their demonstration on the dataset from the University. We further introduce the implemented algorithms that have been used to explain the model decisions and their corresponding results. We close this section by the identified hypotheses of the model behavior.

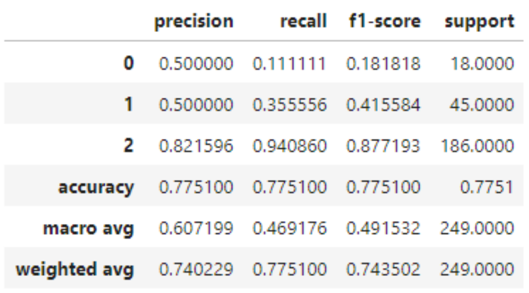
6.3.1 Model Demonstration on the University of Twente Dataset

In the demonstration phase, we took the best performing model and fine-tuned it on the dataset from the University of Twente. According to the results from section 6.2.1 a domain-independent fine-tuning would have been preferable. Due to permission issues only one dataset (i.e. one exam) could be used rather than combining several courses. Therefore, we only fine-tuned the model on one dataset. For the model fine-tuning, we split the dataset into train, validation and test set. Train and test set are split with a ratio of 0.2 in a stratified fashion and the train set is further split into train and validation set with the ratio 0.25. For the model training we used the same pre-trained model checkpoint as the *T5-3B-base* model. Parameters are kept the same with learning rate of 0.001, batch size of 16 and the number of steps with 25,000. The detailed results on the test set can be seen in figure 6.2.

The model performed well on the test data with a weighted average f1-score of 0.7435



(a) Confusion matrix for the test set



	precision	recall	f1-score	support
0	0.500000	0.111111	0.181818	18.0000
1	0.500000	0.355556	0.415584	45.0000
2	0.821596	0.940860	0.877193	186.0000
accuracy	0.775100	0.775100	0.775100	0.7751
macro avg	0.607199	0.469176	0.491532	249.0000
weighted avg	0.740229	0.775100	0.743502	249.0000

(b) Performance evaluation on test set

Figure 6.2: Test set results for university dataset

compared to the validation data with a weighted average f1-score of 0.7146. This higher test score indicates that the model generalizes well on that particular group of answers. However, we observed significant differences between the classes where the model only performs well on correct answers. For incomplete and incorrect answers the model lacks in performance which is a result of the imbalanced nature of the dataset.

This behavior is reflected by an analysis on question level. Here the model performed well on questions that have an imbalanced class distribution where the majority of samples are correct answers. As soon as the class distribution for a question is more balanced we see the reverse pattern where the performance of the model decreases. This behavior is an indicator that the imbalanced nature of the data needs specific consideration and that the model tends to predict wrong answers as correct by default.

6.3.2 Pre-selection of Questions

In the next step, we analyzed each question and the corresponding answers in detail and decided if the question is suitable for a deeper analysis. We ended up selecting the questions 1.1, 5.3, 6.3, 7.1 and 8.3 based on their amount of training samples and their more suitable class distribution which are illustrated in figure 6.3.

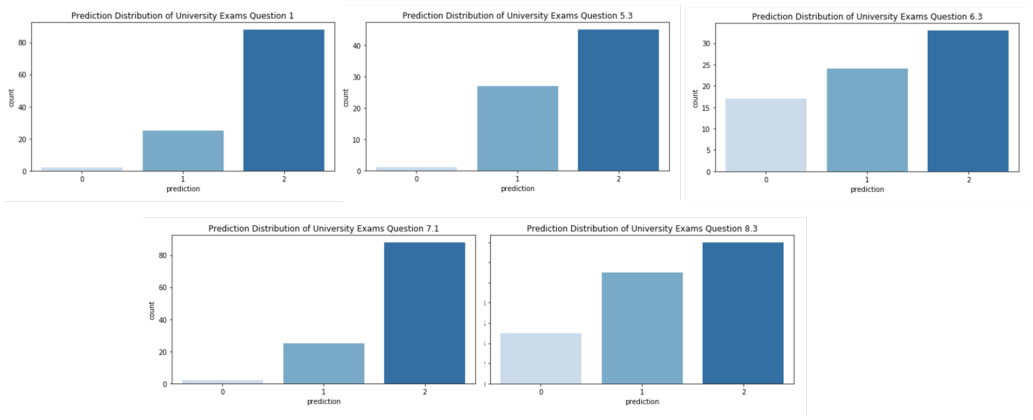


Figure 6.3: Pre-selected label distribution university dataset

6.3.3 First Stage: Analyzing False Positives

In the first stage, we manually analyzed for each question the false positive (i.e. predicted value was 2 but the actual value was less) in order to find specific patterns for inferring model behavior. This analysis was done by skimming through the answers and trying to find commonalities and anomalies. We especially paid attention to similar semantic information in the answers and overlapping words within student responses. We used the model predictions on the test set as well as from the validation set.

The results showed that no clear or significant patterns could be detected that would have allowed a draw a reliable conclusion about model behavior. This is mostly due to the small amount of data in the test and validation set.

6.3.4 Second Stage: Extracting Keywords and Local Key Features

In the second stage, we analyzed the model behavior in greater detail. The central goal of the second stage was to determine the key features for each question and each label in order to derive the main responsible keywords and local key feature for a specific score. For this reason, we applied two different extraction methods.

In the first method, we extracted the keywords separately from each question predicted label using the RAKE Keyword Extraction Method [92]. In detail, we filtered the student answers according to each question and each predicted score. We used the rake-nltk[3] library from python. In order to limit the keywords we only used unigrams, bigrams and trigrams. After applying the algorithm we extracted all keywords with the calculated importance score for each question-score category. In the next step, we manually analyzed the top 10 unigrams, bigrams and trigrams for each category and select the top 5 keyphrases. In addition, we manually searched for answer patterns for each question in the extracted keywords. The results for each score can be seen in the first three rows per question in table 6.4 and table 6.5.

[3]https://pypi.org/project/rake-nltk/

Table 6.4: Identified keywords per method (1/2)

Nr.	Extraction Category	Keyphrases Score 2	Keyphrases Score 1	Keyphrases Score 0
1.1	Trigrams Keyphrase	essentially helps building, entering search terms,country far away, completely replace humans, browse webpages together	-	store might still, size might differ, stock less hassle
1.1	Bigrams Keyphrase	providing instant, physical stores, mortar stores, stores support, retail worker	-	traditional brick, standards related, production shops, offline stores, mortar retailers
1.1	Unigrams Keywords	like, till, service, stores, mortar, customer, shipping	-	stock, used, technology, successful, sometimes, share, product
1.1	L2X Model Keywords	product, online, stores, people, store, shopping, traditional	-	different (2 times) otherwise no overlapping words
5.3	Trigrams Keyphrase	producing outputs shorter, particular market segment, associated services regarding, machine learning algorithems, best option would	named entity recognition, ner involves determining, ner allows one, customer experienced something, help us map	-
5.3	Bigrams Keyphrase	green cars, customer likes, customer group, many reviews, product x	preset groups, customer better, give us, whole context, seperable units	-
5.3	Unigrams Keywords	would, name, machine, like, good, customer, reviews	ner, groups, also, customer, help, words	-
5.3	L2X Model Keywords	product, groups, identify, ner, would, preset, method, used	groups, text, preset, ner, categorizes, sentiment, words	-
6.3	Trigrams Keyphrase	assessing numeric data, recommendations actually work, recall make use, analizing one aspect, normalized absolute error, metrics like precison	last 4 deal, would produce better, precision check whether, better overall image, predict existing ratings	better answer could, advanced calculation techniques, rmse goes back, algorithm performs recall, first groups gives
6.3	Bigrams Keyphrase	first group, algorithm performs, second group, first metrics, nominal data, continuous data, statistical error	first group, second group, distinctive outliers, different since, recommender systems	rmse uses, first group, simplified version, selected elements, rmse removes, rmse measure
6.3	Unigrams Keywords	group, true, second, predict, negative, rmse, recall	precision, better, ratings, group, rmse, performance, measur	rmse, recall, group(s), metrics, directions
6.3	L2X Model Keywords	based, precision, second, metrics, first, ratings	error, group, accuracy, metrics, rmse, first, mse, second	mea, directions, rmse, first, group, mse, remove

Table 6.5: Identified keywords per method (2/2)

Nr.	Extraction Category	Keyphrases Score 2	Keyphrases Score 1	Keyphrases Score 0
7.1	Trigrams Keyphrase	aquisiton per channel, social media influencer, social media advertisment, returning visitors indicates, write complete review	social media referrals, session timing allows, include form submissions, average session duration, social media platforms	-
7.1	Bigrams Keyphrase	social media, bounce rate, highest rate, buyer conversion, conversion rate	search terms, bounce rates, search engine, conversion rate, choose platforms, customer base	-
7.1	Unigrams Keywords	cart, rate, visitors, conversion, user	better, metric, would, want, visitors, visit, user	-
7.1	L2X Model Keywords	product, would, page, know, books, rate, people, website	know, bounce, product, rate, rates, page, check, purchase, improve	-
8.3	Trigrams Keyphrase	one feeds two, free negative marketing, view mindful chef, mindful chef company, added content people, customers already enrolled	one feeds two, limit agricultural chemicals, comparison reflects back, better next time, read positive reviews	-
8.3	Bigrams Keyphrase	mindful chef, hello fresh, negative feedback, good ingredients, good choice, prospective customers	mindful chef, poor children, also help, cempeting providers, good rating	-
8.3	Unigrams Keywords	customers, attention, reputation, charity, readers, hellofresh, blogger	mindful, customer, hellofresh, might, know, attracting	-
8.3	L2X Model Keywords	chef, blogger, know, customers, customers, get, fresh, free	good, additional, providers, receive, help, huge	-

In the second method, we leveraged the model proposed by [61] to detect local key features for each individual answer. In essence, the model used a so called ”instance-wise feature selection method” where the feature selector (CNN architecture) is trained to maximize the mutual information between features and the label. For further technological details please see the paper [61].

We implemented the *L2X for explaining word-based CNN on IMDB* method and adapted the published code to our task. Originally the code was designed to explain a CNN-based binary sentiment classification prediction on the large movie review dataset IMDB[4]. Since our task consists of three possible labels we changed the number of neurons in the explainer-module to 3. Furthermore, we reduced the input sequence from previously 400 to 105 since we deal with short answers. All other parameters are kept the same and can be found in the authors' GitHub repository[5]. Since the model is based on a CNN architecture several pre-processing steps were necessary before the model could be used. First, we created a *word-to-id dictionary* which simply maps each token to a unique number. The unique tokens (i.e. bag-of-words) of the dictionary were created based on the words occurring in question, student answer and reference answer. For these steps we used the word tokenizer from nltk library[6]. Second, we further included start tokens ”$<START>$” which indicate the start of each student answer, unknown words tokens ”$<UNK>$” in case the word does not exist in the dictionary and a padding token ”$<PAD>$” for answers that are shorter than the input sequence of the CNN. In such cases we used pre-padding. After we prepared the dataset, we applied the model and selected the 10 key local features per answer. Since we were interested in the most important tokens we analyzed if local key features were selected by the algorithm more often for answers from a particular question. We did this by grouping the extracted local features by questions and predicted labels and counting the word occurrences. Afterwards depending on the word frequency the final 6-8 keywords per question and score were determined. The results of the local key features for each question and score are illustrated in figure in table 6.4 and 6.5.

6.3.5 Third Stage: Identifying Semantic Relations

The third stage of the analysis focused on the semantic information within the answers for the specific question. We generated semantically related answer cluster and manually identified semantic patterns that trigger the model to classify the answer as correct. In a next step, we combined the previously extracted keyphrases and derived hypotheses which describe the model behavior.

Since an extensive analysis of each answer individually would not be proportionate, we clustered semantically similar answers together and analyzed the clusters as a whole. This made the process viable, more scale-able and allows a better selective analysis of answer groups. The analysis process is divided into four different steps: Extracting answer embed-

[4]https://www.tensorflow.org/datasets/catalog/imdb_reviews

[5]https://github.com/Jianbo-Lab/L2X/tree/master/imdb-word

[6]http://www.nltk.org/_modules/nltk/tokenize.html

dings, filtering each question and score, applying cluster algorithm, inspecting each cluster manually.

Since the fine-tuned model implementation did not provide the possibility to extract the answer-embeddings we used the T5 implementation from Huggingface[7]. Due to memory problems and computation expenses it was only possible to used the pre-trained *t5-large* model for extracting. We extracted the answer-embeddings by averaging the individual word embeddings within a particular answer. These embeddings were then used to generate a distance matrix based on the transformed cosine similarity which is define by:

$$distance(A_i, B_i) = 1 - \cos(\theta) \tag{6.1}$$

Where A_i and B_i represent the distance between two multidimensional answer vectors.

The resulting distance matrix was then used as input for the hierarchical cluster algorithm. For this we used the sklearn library and their Agglomerative Clustering implementation. We chose hierarchical cluster algorithm because of its bottom-up nature and its flexibility in finding a suitable number of cluster. In detail, the algorithm starts by considering each data point (in our case student answer) as a single cluster and merges that with the nearest cluster iteratively. Since we were dealing with data that tends to be inherently different we ensured that we determined a suitable number of cluster. As parameter setting we chose the distance metric *euclidean distance* and as linkage criteria between clusters the *ward method*. The ward method was used because it aims to minimize the overall distance of the data-points within a cluster (i.e. *it minimizes the sum of squared differences*[8]). By that, we ensured that the data-points within a cluster are as similar as possible which eases the later analysis.

After generating the different clusters for each question-score category we investigated each cluster in detail. Within this step, we analyzed the answers for commonalities and answer patterns. Based on the results we derived hypotheses about the model behavior by including the identified keywords and the local features identified in section 6.3.4. The most important result for each of the steps and the derived hypotheses are illustrated in the following section.

6.3.6 Detailed Results and Formulating Model Hypotheses

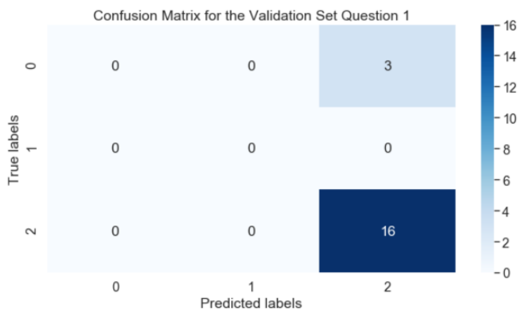
This section contains the results of the five analyzed question and their corresponding derived hypotheses. We show the exact confusion matrix per question for the validation and test set and we report the detailed performance.

Question 1.1

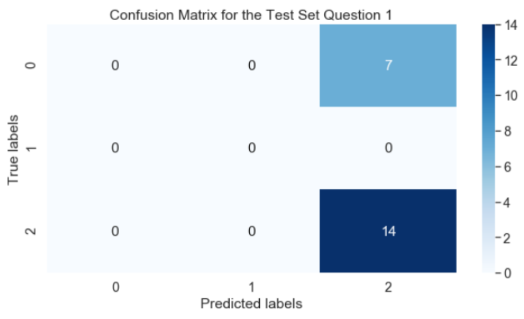
Figure 6.4 shows the confusion matrices for question 1.1 in the University of Twente dataset. The model achieved a weighted average f1-score of 0.5333 for the test set. This score reflects the fact that the model simply predicted all samples as correct. A similar behavior can

[7] https://huggingface.co/transformers/model_doc/t5.html
[8] https://scikit-learn.org/stable/modules/clustering.html

be seen in the performance on the validation set. Here the weighted average f1- score was 0.7699 since the validation set encompasses only three answers with label incorrect. This bias towards correct answer was not a surprise since the model was trained on 69 correct (label 2) and only 7 wrong (label 0) answers.



(a) Confusion matrix for validation set



(b) Confusion matrix for sest set

Figure 6.4: Model results for university dataset question 1.1

After incorporating each of the stages from section 6.3 we derived the following four hypotheses which explain the model behavior for the particular question.

1. Hypothesis 1: Answers that contain the phrases contain consumer, product, store, customer, support, country far away (alone and all together) tend to get full points in their prediction.

2. Hypothesis 2: Answers that contain the phrases product, online, people, store(s), shopping, traditional (alone and all together) tend to get full points in their prediction.

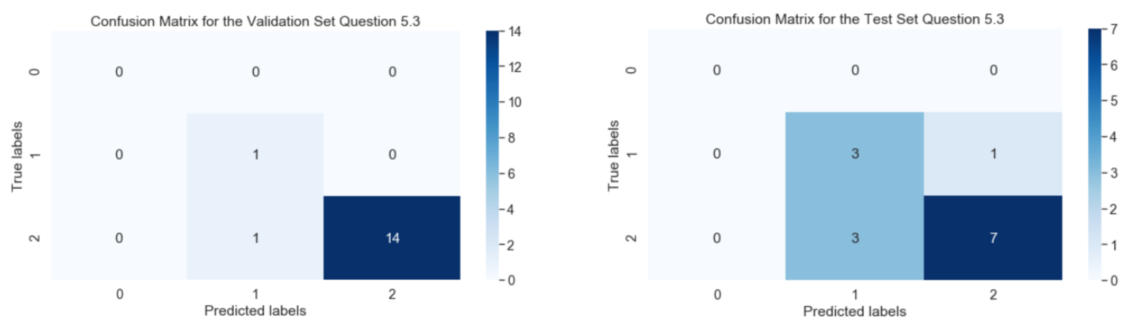
3. Hypothesis 3: Answers that contain vacuous answers but talk about the topic tend to get predicted correct.

4. Hypothesis 4: Answers that contain short and general advantages without being specific tend to get full points in their prediction (e.g. improve customer experience).

Question 5.3

The general model prediction behavior can be seen in the confusion matrices in figure 6.5. The model achieved for the specific question on the test set a weighted average f1-score of 0.7269. For the validation set the weighted average f1- score was 0.9468. This result illustrates clearly the importance of a balanced dataset since it was less biased towards a score. Such model behavior can be traced back to the class distribution of the training set, since the model was trained in a more balanced way on 1 wrong, 19 incomplete and 23 correct answers.

Based on the analysis we derived the following three hypotheses that we think can explain the model behavior.



(a) Confusion matrix for validation set *(b)* Confusion matrix for test set

Figure 6.5: Model results for university dataset question 5.3

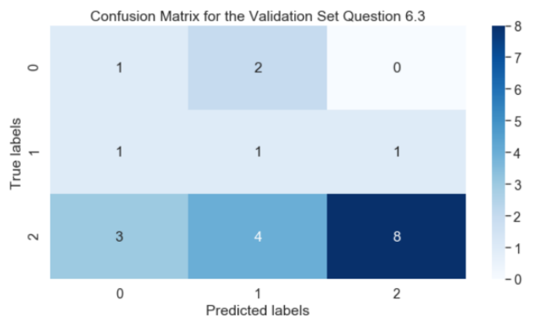
1. Hypothesis 1: Answers that contain the phrases product, groups, identify, preset (alone and all together) tend to get full points in their prediction.
2. Hypothesis 2: Answers that contain the phrases product, machine, customer, reviews, groups, machine learning, algorithm (alone and all together) tend to get full points in their prediction.
3. Hypothesis 3: Answers which start with an general explanation of NER (introducing the general concept of it, correct or even incomplete) combined with a wrong explanation tend to be predicted correct.

Question 6.3

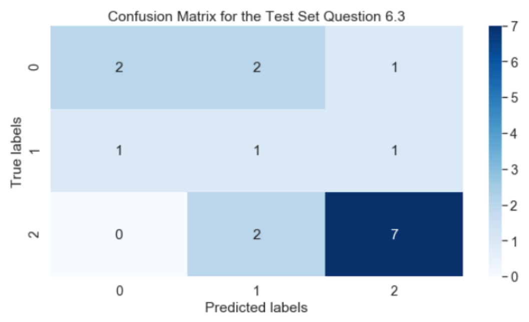
For the third analyzed question, the general model prediction behavior is illustrated in figure 6.6. The model achieved on the test set a weighted average f1-score of 0.6029. For the validation set the weighted average f1- score was 0.5404. The results of the two confusion matrices reflect the indecisiveness of the model in decision making. In contrast to most, the training data was very balanced but the question itself was difficult to answer (e.g. long and nested answers) since naming specific details were of particular importance for the answer. This revealed that complex questions that expect a relatively detailed answer tend to perform worse. That shows that answers, where more complex explanations are expected, the model tends to have difficulties. This makes a correct prediction particularly difficult for longer and complex structured answers.

We derived the following four hypotheses which explain the model behavior for the particular question.

1. Hypothesis 1: Answers that contain the phrases based, precision, ratings (alone and all together) tend to get full points in their prediction.
2. Hypothesis 2: Answers that contain the sub-string ”The MAE,MSE,RMSE predict existing ratings and measure how accurate the algorithm performs.” tend to get full points in their prediction.



(a) Confusion matrix for validation set



(b) Confusion matrix for test set

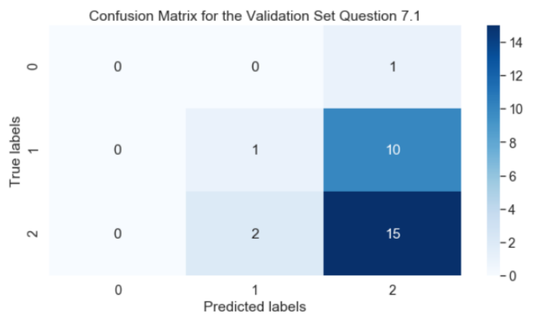
Figure 6.6: Model results for university dataset question 6.3

3. Hypothesis 3: Answers that contain the phrases algorithm performs, nominal data, continuous data, assessing numeric data tend to get full points in the answer.

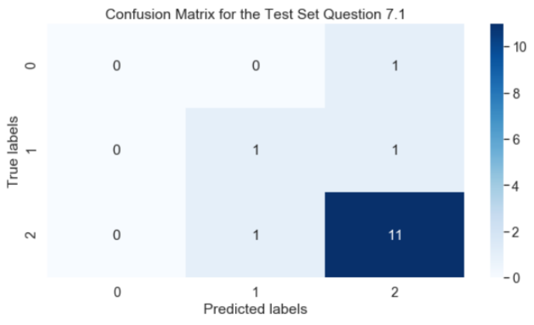
4. Hypothesis 4: General and not specific answers which contain the sub-string ”MAE, MSE, RMSE, are calculated” tend to get predicted correct.

Question 7.1

The general model prediction behavior can be seen in figure 6.7. The model achieved a weighted average f1-score of 0.7706 on the test set. For the validation set the weighted average f1- score was 0.4631. This clearly indicated the model bias towards correct answer, since the test set contained mostly correct answers. This shows again how dependent the model is on the corresponding data structure and class distribution of the training data. In detail, for this specific question the model was trained on 2 wrong, 20 incomplete and 49 correct answers. A deep analysis of the falsely correct (i.e. true class is either 0 or 1 but 2 was predicted) predicted answers revealed that most of the answers were in general correct but fail to include the part of the question which asks for the importance to the business of this feature. Therefore, it seems that the model struggles in distinguishing between complete and incomplete answers and predicts a full score more often because of its bias.



(a) Confusion matrix for validation set



(b) Confusion matrix for test set

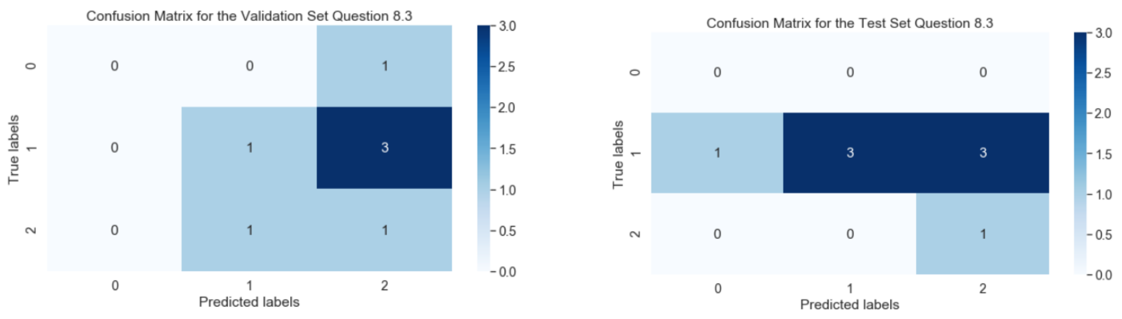
Figure 6.7: Model results for university dataset question 7.1

Based on the analysis we derived four different hypotheses that explain the model behavior.

1. Hypothesis 1: Answers that contain the phrases cart, rate, conversion, conversion rate, buyer conversion (alone and all together) tend to get full points in their prediction.
2. Hypothesis 2: Answers that contain the phrases product, page, books, rate, people, website, insights (alone and all together) tend to get full points in their prediction.
3. Hypothesis 3: Explanations which include many different key metrics terms similar to conversion rate, bounce rate, audience metrics tend to get full points.
4. Hypothesis 4: Incomplete answers that do not provide information about the importance of the mentioned metrics for the business tend to get classified as correct.

Question 8.3

The general model prediction behavior for the last question is illustrated in figure 6.8. For this question, the model achieved a weighted average f1-score of 0.5750 on the test set. For the validation set the weighted average f1- score is 0.2721. The results indicated the randomness in the prediction behavior on the validation set. In addition, it can be seen that the model has a tendency of predicting full scores, although the model learned with more balanced training data. Such a behavior could be accounted to the question type itself since this question asked for the naming of different actors in a particular field. Therefore, the answers had a different answer pattern since most students answered either in bullet-points or coherent sentences. This seemed to make it more difficult for the model since we would expect a relatively good performance. Nevertheless, the small training set does not seem to be enough data to achieve reliable results.



(a) Confusion matrix for validation set *(b)* Confusion matrix for test set

Figure 6.8: Model results for university dataset question 8.3

After the analysis including the extracted keywords and local features, the following hypotheses were developed.

1. Hypothesis 1: Answers that contain the phrases mindful chef often in its explanation (also for explaining other actors) tend to get full points in their prediction.

2. Hypothesis 2: Answers that contain the phrases chef, blogger, customers, good, fresh, hello, charity, riverford tend to get full points in their prediction.

3. Hypothesis 3: Answers that contain the phrases readers, mindful chef, hello fresh, negative feedback, prospective customers, one feeds two, mindful chef company, added content people (alone and together) tend to get full points in their prediction.

6.4 Model Robustness against Adversarial Attacks

In this section, we explain the used methods to generated adversarial answers in order to evaluate the model robustness against student manipulations. First, we analyze to what extend the previously identified hypotheses represent the model behavior. Followed by an evaluation of the models susceptibility towards certain cheating and manipulation strategies.

6.4.1 Answer Creation Process

Based on the determined hypotheses we derived guidelines how the system can be challenged. Herewith we tried to achieve several things. Firstly, the guidelines were formulated in such a way that the hypotheses could either be confirmed or rejected. This made it possible to rule out whether the hypotheses and thus also the approach was successful. Secondly, we aimed for a greater variety of answers by asking different students to trick the system. In addition, the students identified their own manipulation strategies and wrote their own answers. With this approach we ensured that the results are relevant and versatile.

We selected an experimental group of 6 students who were familiar with the topic of the dataset. After we have been instructing the students to answer the questions they went through the answers independently and formulated answers. In order to analyze the reliability and usefulness of the answers, the answers were checked and evaluated by the course professors. If answers were not suitable - meaning that they could be considered partially correct - they were excluded from the validation process of the system. In this step, a total of 20 answers were filtered out, which resulted in a total number of 123 samples. To enable a subsequent analysis of the manipulation strategies we analyzed the answers and grouped cheating strategies to 14 categories. These are shown in table 6.6. A description for each of the strategies is given in the appendix table A.4.

The remaining answers were added as input to the already trained model and the predictions were analyzed accordingly. From the results we deduced to what extent the defined hypotheses correspond to reality and to what extent and with which strategies the system can be tricked. This was done by simply analysing how many of the test answers were predicted correct or partially correct from the model. Based on this number we concluded if the cheating strategy were successful or not.

Table 6.6: Cheating categories and resulting model predictions

Nr.	Cheating Category	Prediction		
		Correct	Incomplete	Wrong
1	Keywords, meaningless answer	37	4	3
2	Keywords, general, unrelated answer	31	5	-
3	General answer, meaningless argument	3	1	
4	Derive solution from question	2	5	-
5	Keywords, general, partially true argument but no explanation	4	2	2
6	General answer, applicable to many things	5	-	-
7	Confusing answer, mixing knowledge 1	3	-	
8	General answer, partially true argument but no explanation	4	2	-
9	Ambiguous answer	2	-	-
10	Correct definition of concept but no valid reason	1	1	-
11	Correct answer to opposite question	1	-	-

6.4.2 Results for each Cheating Strategy

In table 6.6 we reported the aggregated results of the respective model predictions for each of the categories. The detailed results for each individual question can be seen in table 6.7.

It is obvious that when keyphrases were used in the answer, the model tends to classify the answer as correct. However, it should be noted that the cheating strategies were generally rather one-sided and keywords were used in most cases. Furthermore, the detailed results in table 6.7 show different behaviors between the questions. This suggests that a detailed examination of the question/answer types and -structure could provide important insights. We further observed that for relatively general questions (e.g. question 1.1 which asked for a simple advantage) it seems that very general answers regardless of their meaning were classified as correct. However, if the question is more specific - such as questions 5.3, 6.3, 7.1 - the model showed a tendency to classify answers as incomplete or even occasionally as wrong. Especially for general answers and answers that were derived from the question the model was able to classify them as incomplete or even wrong.

Table 6.7: Cheating categories of model robustness test per question

Nr.	Cheating Category	Prediction		
		Correct	Incomplete	Wrong
1.1	Ambiguous answer	2	0	0
	Correct answer to opposite question	1	0	0
	General answer	2	0	0
	General answer, applicable to many things	5	0	0
	General answer, meaningless argument	1	0	0
	Keywords, general, unrelated answer	11	0	0
	Keywords, meaningless answer	7	0	0
5.3	Confusing answer, mixing knowledge	0	1	0
	Derive solution from question	1	4	0
	General answer, meaningless argument	1	0	0
	Keywords, general, partially true argument but no explanation	4	1	0
	Keywords, general, unrelated answer	11	3	0
6.3	Derive solution from question	1	1	0
	Keywords, general, partially true argument but no explanation	0	0	1
	Keywords, general, unrelated answer	4	2	0
	Keywords, meaningless answer	12	3	3
7.1	Confusing answer, mixing knowledge	1	2	0
	Correct definition of concept but no valid reason	1	1	0
	General answer, meaningless argument	0	3	1
	General answer, partially true argument no explanation	1	2	0
	Keywords, general, partially true argument but no explanation	0	1	0
	Keywords, general, unrelated answer	4	0	0
	Keywords, meaningless answer	11	1	0
8.3	General answer, meaningless argument	3	0	0
	Keywords, general, partially true argument but no explanation	2	0	0
	Keywords, general, unrelated answer	1	0	0
	Keywords, meaningless answer	7	0	0

Chapter 7

Discussion and Limitations

In this section, we take up the results from the previous chapter and discuss them for each experiment. The chapter is structured in the same chronological order as the research methodology but with a focus on a more general interpretation of the results and their corresponding implications. In addition, several limitations from the work are mentioned.

7.1 Multi-task Training

We presented an example of transfer learning methods of the T5 model to Automatic Short Answer Grading. With the T5 model that has been trained using a customized dataset selection strategy we obtained mediocre performance improvements. Overall, however it showed the benefits of multi-task learning for ASAG. Significant improvement has been achieved by an extensively pre-trained T5 model that outperformed all previous published models on the ASAG task. This indicates that an extensively trained model achieves better results and that individualized training methods cannot keep up with excessive training. However, the reasons behind these differences are not clear. In addition, the best model achieved similar performance on the category unseen domain and unseen question. This illustrates that the model is not dependent on the domain itself which subsequently means that it either already gained knowledge in the pre-training or that the dataset domains were similar. These findings emphasized the possibilities associated with transfer learning and the multi-task training approach for ASAG. Such a finding aligns with recent developments in NLP research where more extensively trained models to obtain better results.

Despite the design of a suitable model, the performed experiment cannot clearly prove whether the improvements are due to the multi-task training method or to the extensive pre-training. This makes it unclear which design choice for such a model is beneficial. Independent of this, by using such a multi-task training method any relevant NLP tasks can easily be integrated. This ultimately increases the performance and gives educational institutions the freedom to use different tasks in one model. This reduces the data collection effort since the structure of existing exam questions is inherently different.

One of the biggest concerns of the T5 model is the weakness in classifying critical (partially true) answers. It is essential for a model to deal with this particular problem in ASAG since often the accuracy of an answers depends on specific details. Such details could not be detected reliably with our model because we think that it cannot correctly determine the pattern for correct and incomplete answers. This emphasizes that a teacher who is familiar with the details is still necessary for a reliable evaluation and that such an application cannot be used autonomously at present. Nevertheless, it is interesting how the pre-trained model could achieve new-state-of-the-art for the task. This illustrates that even for such difficult tasks as ASAG, development does not stop and transfer learning seems to be a crucial part for further improvements. It makes clear that newly released model architecture should be tested regularly to track progress in the field of ASAG.

7.2 Domain-specific Multi-task Training

Within the experiment, we showed that in general a domain-independent T5 model is preferable to a domain-specific fine-tuned model. Interestingly, the results for both methods show considerable variation within the different domains. This indicates that the questions and answers within the domain seem to be from different difficulties for the models and that the training data is of importance. We believe that the imbalanced nature and the small dataset have significantly influenced the result. For some domains the results indicated a convergence of the performance of the two models when the domain training data increases in size. This could mean that the domain-specific dataset size plays a important role and could flip around the conclusion when a threshold is exceeded. As a consequence, domain-specific fine-tuning would provide a way to compensate for data sparse data. However, since this pattern is only observed in a few cases such an interpretation must be treated with caution and its exact origin and validity must be analyzed more closely.

For the answers from unseen questions, the performance between the methods varies greatly and the non-domain model performs significantly better. From this behavior we can conclude that the non-domain performance benefit from the training of different domain questions. This indicates that there has been a transfer of knowledge between the questions which improved the model generalization capabilities. Such behavior reflects to some extent the natural learning behavior of humans. However, the nature of this behavior is not obvious and it is not clear if this is only due to the amount of training data or whether the domain-specific model overfits the training data.

Furthermore, the results should be considered with regard to the size of the test set. This coupled with the imbalanced dataset dilutes the general significance of the interpretations.

The result, that domain-specific training does not necessarily lead to performance improvements contradicts with some literature that sees potential improvements through domain adaptation in NLP tasks. However, some authors like [27] mitigated this problem by including both domain-specific knowledge and general knowledge to solve the task and achieved good results. This combination seems to be preferable since the generalization ability of the model suffers when focusing purely on a specific domain.

As a consequence, educational institutions do not necessarily have to focus on collecting domain-specific data and can improve the performance by training the model with as much data as possible regardless of the domain. Consequently, this domain-independent fine-tuning leads to a simplification in the collection of usable training data. In addition, it offers the possibility to include any domain, which makes collaboration and data sharing between institutions much easier.

7.3 Model Explainability and Interpretability

Demonstrating the optimized T5 model for the ASAG task resulted in a well-performing but highly biased model. This proved how important an interpretation and analysis of the model decisions is with sparse and imbalanced data.

Four points should be especially considered here. Firstly, the dependency of a model and the importance of a well-balanced dataset. Questions with an imbalanced class distribution achieved relatively good results whereas a more balanced distribution lead to worse results. Such a behavior can be traced back to the imbalanced training which inevitably leads to a preferred prediction of the maturity class. This distorts the interpretation and must therefore be considered differently. Therefore, a simple comparison of performance measures is misleading for this problem, especially when the dataset is so small and a deeper analysis is necessary. For a real implementation, this means that a focus on a balanced dataset is essential for the model usability. Furthermore, the evaluation metrics should be chosen according to the individual requirements combined with an investigation of the performance for each class for understanding and evaluate the model behavior.

Secondly, the importance of differentiated consideration of questions with different degrees of difficulty. The model performance varies with the expected difficulty of given answers. Especially for nested and complex structured answers, the model has severe problems. For such cases it is inevitable to extend the dataset so the model can learn to understand the different semantics patterns that determine a true prediction. The transferred model itself was not able to be adapted perfectly on the given task and did not make up for the data sparse nature. In practice this means that we believe that the more answers the model is trained on the better it becomes. In terms of difficult and specific questions, we think it is advisable to work with several reference answers that capture these specific requirements or to keep the questions simple.

Lastly, the challenges with non-coherent sentences. In some cases incomplete sentences and bullet-points of rather simple questions caused problems for the model. The same applies to answers that are written by using the wrong vocabulary due to language issues. Not surprisingly the model fails to understand the true intended meaning of the answer. Such problems are ubiquitous when working with real exams, since the students are under time pressure and have only average language skills. For such answers it is difficult to rely on this type of a model since it was trained with a different objective. Therefore, we think that such answers have to be taken care of in advance. One possible way would be to educate the student in answering the questions with complete sentences.

For the overall interpretation process, the selected algorithms and the scientific approach to develop behavior hypotheses is advisable for complex models. The incorporation of keywords, local features and the semantic information lead to useful interpretations of the model predictions. Nonetheless, this manual approach is difficult to scale and only feasible with small data size. This creates a conflict of objectives since we believe that the performance can be increase with more data. In such cases, the chosen approach reaches its limits and needs to be replaced with more automated analysis methods.

7.4 Model Robustness against Adversarial Attacks

Evaluation of the demonstrated model showed how susceptible the model is towards manipulations. The majority of the hypothesis-based answers were predicted as correct answers. For this reason, it can be concluded that the hypotheses concerning the predictive behavior of the model were largely confirmed. This makes it particularly prone to manipulation and makes its application in practice as such critical. However, in order to rule out that this result is only due to the imbalanced training method, further experiments and a significant expansion of the dataset is required.

A similar behavior was observed with most of the cheating strategies. It is obvious that the model follows its normal prediction patterns and predicts in most cases the majority class. This is to be interpreted very critically because it means that the model does not understand when an answer is not only wrong but also syntactically completely meaningless. Such behavior can either be attributed to the general tendency of the model to classify responses as correct because of the small and imbalanced dataset, or to the fact that the model indeed lacks in the ability to understand semantic information of the answers. Regardless of the result, these findings emphasize the need to test models for their robustness before deployment.

Nevertheless, in a minority of the cases, the model was able to classify the answer as wrong or as incomplete. This indicates that apart from the general tendency to classify answers as correct, the model shows potential to detect individual manipulation strategies of students. It is interesting in this context that these question types have similar characteristics. They ask for a specific answer that is less open to interpretation and which tends to have a more limited number of possible solutions. This coupled with the fact that the class distributions of these question were more balanced than the others emphasizes again the importance of the dataset for a working model. Therefore, it also seems to make sense to deal with the questions itself in terms of content and to analyze question properties (e.g. question type, difficulty, length, complexity, nesting).

7.5 Limitations

There are also few limitations associated with the experiment which are listed below.

- The imbalanced nature of each of the datasets is one of the main limitations of this work and might dilute some of the results and interpretations.
- For none of the models a hyper-parameter analysis was performed. However, this could lead to different performances of the models and change the results.
- In each of the fine-tuning processes, all model parameters (i.e. weights) were updated. However, other fine-tuning methods may be advantageous, e.g. only the last layers of the model are updated.
- The composition and usage of the different datasets during multi-task pre-training is a limitation because there were great imbalances. This could result in the model being over-trained for a special task that affect the performance.
- The multi-task training ratio was not investigated in this work but is an important factor of multi-task training and can have a significant influence on the model performance.
- Each of the trained models randomly mixed the training samples to assemble the training batches. This can lead to different results per model training which has not been verified in this work.
- The origin of the differences between the domain- and the non-domain-specific is not completely clear. It could also be due to the fact that the questions and answers only partially use specific vocabulary and therefore often have generally formulated answers since the questions are on school level. It might be possible that much more specific questions that go deeper into a domain and therefore require an extended technical vocabulary will benefit from domain-specific training. However, this requires a detailed analysis of the dataset and a professional evaluation.
- Due to a lack of permission only one dataset could be used rather than multiple. This could have influenced the performance of the demonstrated model.
- The efficiency and usefulness of the keywords/local feature extraction methods were not evaluated.
- Some of the reference answers for the University of Twente dataset were selected by the author himself based on his knowledge, research and correct student responses. After validation with a participating professor, some of the reference answers were found to be arguably not completely correct. This could have influenced the model predictions.
- The performed model robustness analysis had only a small-scale character. Therefore not all different strategies could be tested extensively.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

The conclusions is structured in three parts. In the first part, we directly answer the research question that was formulated in Chapter 1. Furthermore, we state to what extent the research goal has been achieved. Part two explains the contributions and implications for the education sector and what institutions can take away from the present work. Lastly, we discuss the involvement of the work to research and illustrate how the present paper makes a valuable contribution to the field of ASAG.

8.1.1 Answer to Research Questions

The objective of this research was to design an ASAG model and evaluate its applicability to practice in order to advance the fields of ASAG and facilitate progress for a digital tutor application. Based on this, the goal was defined in the form of the four main research questions. The answers to these questions are the result of the present research.

1. Research Question: Does multi-task learning improve the performance of Automatic Short Answer Grading?

The research has shown that the multi-task training approach with the T5 model is beneficial for Automatic Short Answer Grading by setting a new-state-of-the-art model for the SciEnts-Bank dataset. Although the results are subject to some uncertainty, they provide sufficient evidence to conclude that multi-task training improves performance in short answer grading.

1. Subquestion 1: Is a multi-task learning approach beneficial when incorporating datasets from the same and related research fields?

   The conducted experiments revealed evidence that training a model with multiple tasks from the same and related fields can indeed improve the performance. The experiments showed that multi-task training leads to an 7.00% increase (weighted average f1 score of 60.69%) in the performance of unseen answers compared to a non-mulit-task trained model. At the same time, this also resulted in a significant decrease in performance in the categories unseen question (-18.20%) and unseen domain (-12.69%). Furthermore, with the chosen approach it was not possible to outperform the defined baseline for the ScieEntsBank dataset. However, it is not clear from the results to what extent multi-task training is responsible or the improvement on unseen answer. A more detailed discussion about this can be seen in section 7.4.

2. Subquestion 2: Does a mulit-task pre-trained model improve Automatic Short Answer Grading and outperform the baseline?

   The results showed that by using an extensively pre-trained model a new state-of-the-art performance was achieved for the SciEntsBank dataset. The chosen model outperformed the baseline by long shot. For the category unseen answer it achieved a weighted average f1-score of 74.51% (+7.31%), for unseen questions 66.87% (+13.77%) and for unseen domains 67.88% (+10.48%). Therefore the results support the fact that pre-trained models are clearly preferable to individually trained models and it does not seem to matter which NLP tasks are used. This emphasizes the significant role of the extent of the training and the complexity of the model.

2. Research Question: Does domain-specific fine-tuning influence the performance of Automatic Short Answer Grading?

It was found that the non-domain model outperformed the domain-specific model in each of the 12 domains and each question category. This showed that non-domain models benefit from the training of different domains. According to the results, this can be traced back to the fact that the generic learning model uses the knowledge gained from different domains. Whereas the domain-specific model lacks in generalization capabilities.

3. Research Question: How can we explain model decisions in a real-world application?

Within this work, we developed a framework for the analysis of model behavior of a real-world exam dataset from the University of Twente. The model achieved a weighted average f1-score of 74.35% on the test set. With the identification of keywords and local answer features (RAKE method and L2X model) and the application of hierarchical cluster algorithm different prediction patterns in the answers could be identified. Based on the patterns we were able to formulate suitable hypotheses for selected questions that describe the model behavior and explain their decisions. During the analysis some important insights were detected. It turned out that the high test score should be viewed with extreme caution since the model as a whole predominantly classifies answers as correct. This is due to the characteristics of the training set, in which there is a clear imbalance which results in a bias towards correct answers. Such enormous imbalances in the data set and the additional small amount of data presented a great challenge and diluted the analysis.

4. Research Question: How robust is the model towards student manipulations?

One of the most neglected issues in ASAG was brought into focus with this question. Based on the analysis of the model behavior, adversarial attacks were created. These were supplemented by selected students who tried to trick the system for specific questions. These attempts were combined into cheating strategies and clearly showed how susceptible the model is towards manipulation attempts. In particular, answers that contain the identified keywords and were either completely meaningless or unrelated to the actual question were mostly able to get a full score. Furthermore, there was a slight correlation between the difficulty of the question and the degree of susceptibility. In some cases, answers to more specific questions could be classified as partially correct or incorrect. Therefore the evaluated model is not suitable for practical use under these conditions.

Achievement of the Research Goal

The work showed that an efficient model could be developed which sets a new benchmark for the target dataset. However, when applied to the university context, it shows clear weaknesses. The main reasons for the weaknesses were the small number of training samples coupled with imbalanced training data. The model was not able to deal with these efficiently. The decision bases of the model could be identified with suitable algorithms. However, the results were diluted by the poor data basis. Furthermore, it was clearly shown how susceptible the model is towards student manipulations and how strong the dependency on the training data was.

Regarding the progress of a digital tutor, we can say that we have not yet arrived at a practical model for a digital tutor. The main reason for poor performance is the difficulty of the task itself and the data situation. Currently, a very well-performing transformer model is not able to give a human-like evaluation. Nevertheless, the results suggest that we are on the right track and that researcher should work on the data basis of ASAG.

8.1.2 Educational Implications and Contribution to Practice

This work was motivated by the goal of developing a practical solution or at least advance the field with a practical application focus. The present research therefore, offers some contributions to practice and implications for educational institutions.

First, the model shows that domain-specific training is not beneficial but rather the amount of training data is from importance. This means that institutions do not necessarily have to collect domain-specific data. This allows for extensive cooperation among each other, which can lead to data sharing models and collaborative advances. Second, the used model shines by its simple integration possibility of different tasks. Thus the use of the T5 architecture offers enormous potential and can be developed further by simply preparing new data sets. This makes usage and continuous expansion possible without adapting the model, which is an advantage in today's variety of questions. Third, due to the superiority of a pre-trained model, this facilitates the application of the model as no extensive training from scratch is required. This saves valuable resources and makes it more environmentally friendly. Nevertheless, it should be noted that the size of the model should not be underestimated and the necessary infrastructure must be guaranteed for such applications in future. Fourth, the framework for an interpretation of the results can be used and adapted to the respective requirements. However, the clustering of exams according to semantic similarity is hardly scalable. Here the manual approach reaches its limits after a certain number of questions/answers pairs. Fifth, the research provides an incentive to take a closer look at the question/answer structures and the degree of difficulty for such a model application. This might lead to the development of new question structures and guidelines. Finally, one of the most important implications should be mentioned. Educational institutions should be realistic and critical about ASAG because there is still a lot of potentials. The robustness test provided an incentive to take a critical look at the models and not to rely on them blindly. Furthermore, autonomous and accurate applications that replace humans are not yet feasible.

8.1.3 Contribution to Research

The contributions to research through comprehensive work is versatile. First, the application of the multi-task training approach to the problem of ASAG showed successful feasibility. To the best of our knowledge, such an approach has not been done before. Within the experiments, a new state-of-the-art model was created which outperforms the previous models significantly and therefore a new milestone for this dataset was reached. This combination lays the foundation for a more focused scientific discussion on multi-task learning models in the field of ASAG. Secondly, the obtained results regarding the advantage of generically trained models over domain-specific ones stimulate an extensive discussion about the training methodology of such models. Third, the research applied a model that performs very well on a ASAG dataset to a real exam of a university. This allowed to analyze the model more precisely and to identify the basis for decision making. Therefore a scientific approach using hypotheses was presented based on a compounded and in-depth model-agnostic analysis. This approach can be used by other researchers to make progress in the field of model explainability. It could also be shown that an exact examination of the respective performance metrics is necessary and that especially the dataset should be considered in detail. Fourth, by challenging the algorithm, the weaknesses of such a system could be highlighted. The identification of the far-reaching influence of an imbalanced dataset leads to an incentive that such models can not only be used for popular datasets but are also resilient enough to be suitable for practical application. For this purpose, a further applicable methodology was developed which can be further refined and developed. Finally, the proposed work combines the most important features of a model which are necessary for a practical application. It supports researchers in approaching this problem as a multifaceted one with a practical focus. In addition, it creates awareness for model explainability and resilience which have been neglected by researchers. Individual features of this work can be further developed by researchers taking into account the important factors. This leads to a new research approach in the field of ASAG which aims at a practical solution.

8.2 Recommendations and Future Work

The recommendations and possible future research are divided into two parts. Here we show to what extent the present work can be extended and improved.

8.2.1 Multi-task Training and Domain-specific Fine-tuning

The presented model training and fine-tuning could be improved and extended in several ways.

As discussed above, the research did not provide absolute evidence of the superiority of multi-task training. Therefore a next step would be to transform the multiple tasks to an unsupervised problem and train the model only on textual data. The performance can than be compared to the already trained model which provides enough evidence for a reliable conclusion. In addition, it could be analyzed how the hyper-parameter choice influences the results and whether the conclusions is preserved. Especially the experiments 2-3 could be repeated with different hyper-parameters (especially batch size, learning rate and step size). This could be used to improve performance and inspect why the model (experiment 2) showed such a performance drop in its generalization ability.

Due to the random composition of training and fine-tune batches, the implementation could be adapted so that the models are trained with exactly the same batches. This would help to reduce the diluted results of the experiments. However, additional pre-training on large amount of data is associated with enormous computational effort. When experimenting with such large models it should not be the incentive to develop always exclusively better models, but rather the efficiency and resource conservation should be considered. Therefore, one should be weighing between necessity and sustainability before training a new model. Based on the experience of the work, we advise not to train extensive models on your own but to use pre-trained models and further train them.

We further recommend enhancing the adaptation process by analyzing if another fine-tune process is beneficial. One possibility would be to experiment in adjusting the number of parameters (e.g. only adapt last few layers) that are updated during fine-tuning. This could lead to better performance and to a reduction of the training time. In addition, a further step would be to conduct a similar domain adaptation experiment but keeping the fine-tuning datasets the same size. The reason for this is that it can be excluded whether the larger data volume of the generic model is responsible for the better performance. We expect that this could solve the differences between the dataset splits within a domain. However, this requires a careful combination of different datasets or even the creation of a new one.

We also believe that we would have obtained even better results by using the largest 11B model. This would improve the performance even more, but also increase the computational costs and the model could only be trained with special TPUs from Google.

8.2.2 Model Explainability and Robustness

As mentioned data scarcity is one of the major obstacles. For the demonstration and evaluation we recommend performing the experiments in the field of model explainability and robustness with a balanced and comprehensive dataset. This can be achieved by collecting more comprehensive data from exams or student tasks. Especially in the digital age we expect a growth in availability of data. Furthermore, various ways to deal with the class imbalances like data amplification could be investigated. For example the feasibility and application of classical methods like [93], [96], [94], [97] to such a NLP problem could be analyzed.

Studying ways to incorporate model-agnostic and non-agnostic interpretation approaches (like attention weights visualizations) is a great challenge but could produce interesting findings. Likewise, the key-phrase extraction methods could be analyzed in greater detail and the accuracy could be validated. By applying the two methods to a balanced dataset one could make valuable inferences to the importance of keywords and local key features. With this more reliable hypotheses could be found and tested. In addition, it could be analyzed if more scalable methods are applicable to cluster the answers based on semantic information and content. The goal should be to automatically identify specific topics of the answers. An interesting approach could be to analyze answers with topic model or similar approaches.

Further, researchers could gradually increase the number of labels in order to increase the difficulty of the task. By that, the model learns to distinguish between correct and partially correct answers since this is one the main lacks which prevents current autonomous application. An analysis of how advanced training can improve model capabilities is crucial for a real application. Such training could be complemented by integrating different manipulation responses which specifically influence the model training. Thereby we expect that the mentioned weaknesses of the model can be reduced. For this purpose, specific answers can be formulated which are continuously added to the model training to improve the performance and resilience towards cheating strategies.

Bibliography

[1] S. Ruder, “Neural transfer learning for natural language processing,” Ph.D. dissertation, National University of Ireland, Galway, 2019.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *ArXiv*, vol. abs/1706.03762, 2017.

[3] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *ArXiv*, vol. abs/1910.10683, 2019.

[4] N. Suzen, A. N. Gorban, J. Levesley, and E. M. Mirkes, “Automatic short answer grading and feedback using text mining methods,” *ArXiv*, vol. abs/1807.10543, 2018.

[5] S. Burrows, I. Gurevych, and B. Stein, “The eras and trends of automatic short answer grading,” *International Journal of Artificial Intelligence in Education*, vol. 25, pp. 60–117, 2014.

[6] R. Collobert and J. Weston, “A unified architecture for natural language processing: deep neural networks with multitask learning,” in *ICML ’08*, 2008.

[7] L. Deng, G. E. Hinton, and B. Kingsbury, “New types of deep neural network learning for speech recognition and related applications: an overview,” *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 8599–8603, 2013.

[8] R. B. Girshick, “Fast r-cnn,” *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448, 2015.

[9] R. Caruana, “Multitask learning,” in *Encyclopedia of Machine Learning and Data Mining*, 1998.

[10] M. Hightower. (2020) High-level history of nlp models. [Online]. Available: https://towardsdatascience.com/high-level-history-of-nlp-models-bc8c8b142ef7

[11] S. Saha, T. I. Dhamecha, S. Marvaniya, P. Foltz, R. Sindhgatta, and B. Sengupta, “Joint multi-domain learning for automatic short answer grading,” *ArXiv*, vol. abs/1902.09183, 2019.

[12] B. Riordan, A. Horbach, A. Cahill, T. Zesch, and C. M. Lee, “Investigating neural architectures for short answer scoring,” in *BEA@EMNLP*, 2017.

[13] E. Ma. (2020) Text-to-text transfer transformer. [Online]. Available: https://medium.com/dataseries/text-to-text-transfer-transformer-e35dc28bae14

[14] A. Roberts. (2020) Exploring transfer learning with t5: the text-to-text transfer transformer. [Online]. Available: https://ai.googleblog.com/2020/02/exploring-transfer-learning-with-t5.html

[15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *ArXiv*, vol. abs/1810.04805, 2019.

[16] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in *NeurIPS*, 2019.

[17] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," *ArXiv*, vol. abs/1808.06226, 2018.

[18] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," *ArXiv*, vol. abs/1508.07909, 2016.

[19] T. Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," in *ACL*, 2018.

[20] J. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *ArXiv*, vol. abs/1607.06450, 2016.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[22] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, 2014.

[23] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-attention with relative position representations," in *NAACL-HLT*, 2018.

[24] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, I. Simon, C. Hawthorne, N. Shazeer, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, "Music transformer: Generating music with long-term structure," in *ICLR*, 2019.

[25] N. Shazeer and M. Stern, "Adafactor: Adaptive learning rates with sublinear memory cost," in *ICML*, 2018.

[26] C. Sung, T. I. Dhamecha, and N. Mukhi, "Improving short answer grading using transformer-based pre-training," in *AIED*, 2019.

[27] S. Roy, H. S. Bhatt, and Y. Narahari, "An iterative transfer learning based ensemble technique for automatic short answer grading," *ArXiv*, vol. abs/1609.04909, 2016.

[28] A. Magooda, M. A. Zahran, M. Rashwan, H. M. Raafat, and M. B. Fayek, “Vector based techniques for short answer grading,” in *FLAIRS Conference*, 2016.

[29] M. A. Sultan, C. Salazar, and T. Sumner, “Fast and easy short answer grading with high accuracy,” in *HLT-NAACL*, 2016.

[30] C. Sung, T. I. Dhamecha, S. Saha, T. Ma, V. P. Reddy, and R. Arora, “Pre-training bert on domain resources for short answer grading,” in *EMNLP/IJCNLP*, 2019.

[31] M. O. Dzikovska, R. D. Nielsen, C. Brew, C. Leacock, D. Giampiccolo, L. Bentivogli, P. Clark, I. Dagan, and H. T. Dang, “Semeval-2013 task 7: The joint student response analysis and 8th recognizing textual entailment challenge,” in *SemEval@NAACL-HLT*, 2013.

[32] M. Mohler, R. C. Bunescu, and R. Mihalcea, “Learning to grade short answer questions using semantic similarity measures and dependency graph alignments,” in *ACL*, 2011.

[33] H. Foundation. (2020) The hewlett foundation: Short answer scoring. [Online]. Available: https://www.kaggle.com/c/asap-sas/overview

[34] L. B. Galhardi, H. C. de Mattos Senefonte, R. C. T. de Souza, and J. D. Brancher, “Exploring distinct features for automatic short answer grading,” 2018.

[35] S. Ruder, “An overview of multi-task learning in deep neural networks,” *ArXiv*, vol. abs/1706.05098, 2017.

[36] Y. Zhang and Q. Yang, “A survey on multi-task learning,” *ArXiv*, vol. abs/1707.08114, 2017.

[37] K. Hashimoto, C. Xiong, Y. Tsuruoka, and R. Socher, “A joint many-task model: Growing a neural network for multiple nlp tasks,” in *EMNLP*, 2017.

[38] V. Sanh, T. Wolf, and S. Ruder, “A hierarchical multi-task approach for learning embeddings from semantic tasks,” *ArXiv*, vol. abs/1811.06031, 2019.

[39] R. French, “Catastrophic forgetting in connectionist networks,” *Trends in Cognitive Sciences*, vol. 3, pp. 128–135, 1999.

[40] X. Liu, P. He, W. Chen, and J. Gao, “Multi-task deep neural networks for natural language understanding,” *ArXiv*, vol. abs/1901.11504, 2019.

[41] J. Niehues and E. Cho, “Exploiting linguistic resources for neural machine translation using multi-task learning,” in *WMT*, 2017.

[42] S. Subramanian, A. Trischler, Y. Bengio, and C. Pal, “Learning general purpose distributed sentence representations via large scale multi-task learning,” *ArXiv*, vol. abs/1804.00079, 2018.

[43] E. Kiperwasser and M. Ballesteros, “Scheduled multi-task learning: From syntax to translation,” *Transactions of the Association for Computational Linguistics*, vol. 6, pp. 225–240, 2018.

[44] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith, “Don’t stop pretraining: Adapt language models to domains and tasks,” *ArXiv*, vol. abs/2004.10964, 2020.

[45] I. Beltagy, K. Lo, and A. Cohan, “Scibert: A pretrained language model for scientific text,” in *EMNLP/IJCNLP*, 2019.

[46] D. Araci, “Finbert: Financial sentiment analysis with pre-trained language models,” *ArXiv*, vol. abs/1908.10063, 2019.

[47] J.-S. Lee and J. Hsiang, “Patentbert: Patent classification with fine-tuning a pre-trained bert model,” *ArXiv*, vol. abs/1906.02124, 2019.

[48] S. Meftah, N. Semmar, M.-A. Tahiri, Y. Tamaazousti, H. Essafi, and F. Sadat, “Multi-task supervised pretraining for neural domain adaptation,” in *SOCIALNLP*, 2020.

[49] S. Deena, R. W. M. Ng, P. Madhyastha, L. Specia, and T. Hain, “Semi-supervised adaptation of rnnlms by fine-tuning with domain-specific auxiliary features,” in *INTER-SPEECH*, 2017.

[50] S. S. Jeawak, L. Espinosa-Anke, and S. Schockaert, “Cardiff university at semeval-2020 task 6: Fine-tuning bert for domain-specific definition classification,” 2020.

[51] H. Song, R. Dabre, A. Fujita, and S. Kurohashi, “Domain adaptation of neural machine translation through multistage fine-tuning,” 2020.

[52] M. Dehghani, S. Gouws, O. Vinyals, J. Uszkoreit, and L. Kaiser, “Universal transformers,” *ArXiv*, vol. abs/1807.03819, 2019.

[53] A. Coenen, E. Reif, A. Yuan, B. Kim, A. Pearce, F. Viégas, and M. Wattenberg, “Visualizing and measuring the geometry of bert,” in *NeurIPS*, 2019.

[54] S. Serrano and N. A. Smith, “Is attention interpretable?” in *ACL*, 2019.

[55] S. Jain and B. C. Wallace, “Attention is not explanation,” in *NAACL-HLT*, 2019.

[56] S. Wiegreffe and Y. Pinter, “Attention is not not explanation,” in *EMNLP/IJCNLP*, 2019.

[57] S. Abnar and W. Zuidema, “Quantifying attention flow in transformers,” in *ACL*, 2020.

[58] C. Sen, T. Hartvigsen, B. Yin, X. Kong, and E. A. Rundensteiner, “Human attention maps for text classification: Do humans and neural networks focus on the same words?” in *ACL*, 2020.

[59] M. T. Ribeiro, S. Singh, and C. Guestrin, "”why should i trust you?”: Explaining the predictions of any classifier," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[60] ——, "Anchors: High-precision model-agnostic explanations," in *AAAI*, 2018.

[61] J. Chen, L. Song, M. J. Wainwright, and M. I. Jordan, "Learning to explain: An information-theoretic perspective on model interpretation," *ArXiv*, vol. abs/1802.07814, 2018.

[62] W. Zhang, Q. Z. Sheng, A. Alhazmi, and C. Li, "Adversarial attacks on deep-learning models in natural language processing," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 11, pp. 1 – 41, 2020.

[63] T. He and J. R. Glass, "Detecting egregious responses in neural sequence-to-sequence models," *ArXiv*, vol. abs/1809.04113, 2019.

[64] M. Cheng, J. Yi, H. Zhang, P. Chen, and C.-J. Hsieh, "Seq2sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples," *ArXiv*, vol. abs/1803.01128, 2020.

[65] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, "Hotflip: White-box adversarial examples for text classification," in *ACL*, 2018.

[66] E. Wallace, S. Feng, N. Kandpal, M. Gardner, and S. Singh, "Universal adversarial triggers for attacking and analyzing nlp," in *EMNLP/IJCNLP*, 2019.

[67] M. Behjati, S. Moosavi-Dezfooli, M. S. Baghshah, and P. Frossard, "Universal adversarial attacks on text classifiers," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 7345–7349.

[68] Y. Belinkov and Y. Bisk, "Synthetic and natural noise both break neural machine translation," *ArXiv*, vol. abs/1711.02173, 2018.

[69] R. Jia and P. Liang, "Adversarial examples for evaluating reading comprehension systems," *ArXiv*, vol. abs/1707.07328, 2017.

[70] E. Wallace, P. Rodriguez, S. Feng, I. Yamada, and J. L. Boyd-Graber, "Trick me if you can: Human-in-the-loop generation of adversarial examples for question answering," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 387–401, 2019.

[71] M. T. Ribeiro, T. Wu, C. Guestrin, and S. Singh, "Beyond accuracy: Behavioral testing of nlp models with checklist," in *ACL*, 2020.

[72] F. Yin, Q. Long, T. Meng, and K.-W. Chang, "On the robustness of language encoders against grammatical errors," *ArXiv*, vol. abs/2005.05683, 2020.

[73] R. Marvin and T. Linzen, “Targeted syntactic evaluation of language models,” in *EMNLP*, 2018.

[74] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, “A large annotated corpus for learning natural language inference,” *ArXiv*, vol. abs/1508.05326, 2015.

[75] A. Williams, N. Nangia, and S. R. Bowman, “A broad-coverage challenge corpus for sentence understanding through inference,” *ArXiv*, vol. abs/1704.05426, 2018.

[76] T. Khot, A. Sabharwal, and P. Clark, “Scitail: A textual entailment dataset from science question answering,” in *AAAI*, 2018.

[77] A. S. Gordon, Z. Kozareva, and M. Roemmele, “Semeval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning,” in *SemEval@NAACL-HLT*, 2012.

[78] I. Dagan, O. Glickman, and B. Magnini, “The pascal recognising textual entailment challenge,” in *MLCW*, 2005.

[79] M. Marelli, L. Bentivogli, M. Baroni, R. Bernardi, S. Menini, and R. Zamparelli, “Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment,” in *SemEval@COLING*, 2014.

[80] D. M. Cer, M. T. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia, “Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation,” *ArXiv*, vol. abs/1708.00055, 2017.

[81] W. B. Dolan and C. Brockett, “Automatically constructing a corpus of sentential paraphrases,” in *IWP@IJCNLP*, 2005.

[82] N. D. Shankar Iyer and K. Csernai. (2020) Quora question pairs. [Online]. Available: https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs

[83] P. Rajpurkar, R. Jia, and P. Liang, “Know what you don’t know: Unanswerable questions for squad,” *ArXiv*, vol. abs/1806.03822, 2018.

[84] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, “Squad: 100, 000+ questions for machine comprehension of text,” in *EMNLP*, 2016.

[85] G. Lai, Q. Xie, H. Liu, Y. Yang, and E. Hovy, “Race: Large-scale reading comprehension dataset from examinations,” in *EMNLP*, 2017.

[86] S. Zhang, X. Liu, J. Liu, J. Gao, K. Duh, and B. V. Durme, “Record: Bridging the gap between human and machine commonsense reading comprehension,” *ArXiv*, vol. abs/1810.12885, 2018.

[87] D. Khashabi, S. Chaturvedi, M. Roth, S. Upadhyay, and D. Roth, “Looking beyond the surface: A challenge set for reading comprehension over multiple sentences,” in *NAACL-HLT*, 2018.

[88] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova, “Boolq: Exploring the surprising difficulty of natural yes/no questions,” *ArXiv*, vol. abs/1905.10044, 2019.

[89] R. Navigli, J. Camacho-Collados, and A. Raganato, “Word sense disambiguation: A unified evaluation framework and empirical comparison,” in *EACL*, 2017.

[90] H. J. Levesque, E. Davis, and L. Morgenstern, “The winograd schema challenge,” in *KR*, 2011.

[91] M. T. Pilehvar and J. Camacho-Collados, “Wic: 10, 000 example pairs for evaluating context-sensitive representations,” *ArXiv*, vol. abs/1808.09121, 2018.

[92] S. J. Rose, D. Engel, N. Cramer, and W. Cowley, “Automatic keyword extraction from individual documents,” 2010.

[93] I. Mani and I. Zhang, “kNN approach to unbalanced data distributions: a case study involving information extraction,” in *Proceedings of workshop on learning from imbalanced datasets*, vol. 126, 2003.

[94] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[95] E. Goceri and A. Gooya, “On the importance of batch size for deep learning,” 07 2018.

[96] H. Han, W. Y. Wang, and B. H. Mao, “Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning,” in *Lecture Notes in Computer Science*, vol. 3644, no. PART I. Springer, Berlin, Heidelberg, 2005, pp. 878–887.

[97] H. He, Y. Bai, E. A. Garcia, and S. Li, “ADASYN: Adaptive synthetic sampling approach for imbalanced learning,” in *Proceedings of the International Joint Conference on Neural Networks*, 2008, pp. 1322–1328.

Appendix A

Appendix

A.1 Multi-task Training Preprocessed Inputs

In this section we give an example for each preprocessed input format for the datasets used. For the processed labels the label range is indicated, in the application only the *true label* is processed and not all of them.

COPA

Processed Input: ”copa premise: my body cast a shadow over the grass. choice1: the sun was rising. choice2: the grass was cut. question: cause”
Processed Target Labels: ”true/false”

MSRP

Processed Input: ”mrpc sentence1: amrozi accused his brother, whom he called the witness, of deliberately distorting his evidence. sentence2: referring to him as only the witness, amrozi accused his brother of deliberately distorting his evidence.”
Processed Target Labels: ”true/false”

MultiNLI

Processed Input: ”multinli sentence1: how do you know? all this is their information again. sentence2: this information belongs to them”
Processed Target Labels: ”entailment/neutral/no entailment”

MultiRC

Processed Input: ”multirc passage: while this process moved along, diplomacy continued its rounds. direct pressure on the taliban had proved unsuccessful. as one nsc staff note put it, under the taliban, afghanistan is not so much a state sponsor of terrorism as it is a state sponsored by terrorists. [...] president clinton told us that when he pulled musharraf aside for a brief, one-on-one meeting, he pleaded with the general for help regarding bin laden.”” i offered him the moon when i went to see him, in terms of better relations with the united states, if hed help us get bin laden and deal with another issue or two.”” the u.s. effort

continued. ” question: what did the high-level effort to persuade pakistan include? answer: children, gerd, or dorian popa”
Processed Target Labels: ”true/false”

ReCoRD[1]
Processed Input: ”record passage: the harrowing stories of women and children locked up for so-called moral crimes’ in afghanistan’s notorious female prison have been revealed after cameras were allowed inside. mariam has been in badam bagh prison for three months after she shot a man who just raped her at gunpoint and then turned the weapon on herself - but she has yet to been charged. nuria has eight months left to serve of her sentence for trying to divorce her husband. she gave birth in prison to her son and they share a cell together. scroll down for video nuria was jailed for trying to divorce her husband. her son is one of 62 children living at badam bagh prison most of the 202 badam bagh inmates are jailed for so-called ’moral crimes crimes include leaving their husbands or refusing an arrange marriage 62 children live there and share cells with their mothers and five others query: the baby she gave birth to is her husbands and he has even offered to have the courts set her free if she returns, but ***@placeholder*** has refused.”
Processed Target Labels: ”nuria”

RTE
Processed Input: ”rte premise: no weapons of mass destruction found in iraq yet. hypothesis: weapons of mass destruction found in iraq.”
Processed Target Labels: ”entailment/no entailment”

SNLI
Processed Input: ”snli sentence1: a person on a horse jumps over a broken down airplane. sentence2: a person is training his horse for a competition.”’
Processed Target Labels: ”entailment/neutral/no entailment”

SICK
Processed Input: ”sick sentence1: the young boys are playing outdoors and the man is smiling nearby. sentence2: the kids are playing outdoors near a man with a smile.”
Processed Target Labels: ”entailment/neutral/no entailment”

STS-Benchmark
Processed Input: stsb sentence1: a plane is taking off. sentence2: an air plane is taking off.’, ’targets’: b’5.00’
Processed Target Labels: ”0.00/0.25/0.50/.../4.50/4.75/5.00”

[1]For this datasets the authors used a customized appraoch in order to transform the tasks to the text-to-text input format. We only included samples that have three or less possible answer choices where all have the same string, i.e. are duplicates and refer to the same object or subject

SQuAD1.0

Processed Input: ”squad context: architecturally, the school has a catholic character. atop the main buildingś gold dome is a golden statue of the virgin mary. immediately in front of the main building and facing it, is a copper statue of christ with arms upraised with the legend ”venite ad me omnes”. next to the main building is the basilica of the sacred heart. immediately behind the basilica is the grotto, a marian place of prayer and reflection. it is a replica of the grotto at lourdes, france where the virgin mary reputedly appeared to saint bernadette soubirous in 1858. at the end of the main drive (and in a direct line that connects through 3 statues and the gold dome), is a simple, modern stone statue of mary. question: to whom did the virgin mary allegedly appear in 1858 in lourdes france?”
Processed Target Labels: ”saint bernadette soubirous”

ASAP-SAS

Processed Input: ”asap essayset: 1. essaytext: what you need is more trials, a control set up, and an exact amount of vinegar to pour in each cup/beaker. you could also take and check the mass every 30 min or 1 hour”
Processed Target Labels: ”0/1/2/3”

University of Texas

Processed Input: ”texas question: what is the role of a prototype program in problem solving? reference answer: to simulate the behaviour of portions of the desired software product. answer: a prototype program simulates the behaviors of portions of the desired software product to allow for error checking.”
Processed Target Labels: ”0.0/0.5/1/.../4/4.5/5.0”

WiC

Processed Input: ”wic sentence1: approach a task. sentence2: to approach the city. word: approach”
Processed Target Labels: ”true/false”

WSC

Processed Input: ”wsc text: i poured water from the bottle into the cup until ***it*** was full. word: the cup”
Processed Target Labels: ”true/false”

SciEntsBank

Processed Input: ”semeval question: when conducting a controlled experiment, why do you use a standard? semeval reference answers: a standard is used for comparison to determine how changing one variable changes the results. semeval student answers: to keep the variables controlled.”
Processed Target Labels: ”0/1/2/3/4”

A.2 University of Twente Dataset Preprocessed Inputs

Processed Input: ”exams_question: What advantages do traditional brick and mortar retailers have over e-commerce only retailers when it comes to latest developments in e-commerce? exams_reference_answers: they already have an offline shop infrastructure in place which enables them to establish a cross-channel experience. exams_student_answers: groceries are the most in-store bought items. by expanding from online stores to physical stores amazon can expand their market influence.”
Processed Target Labels: ”0/1/2”

A.3 Domain Adaptation and Model Robustness

Table A.1: Parameters for model training

Domain	Batch Size	Step Size	Epochs
EM	8	4,111	115
FN	8	4,111	115
II	8	2,041	115
LF	8	3,766	115
LP	8	661	115
ME	8	7,935	115
MS	8	2,415	115
MX	8	6,670	115
PS	8	5,218	115
SE	8	5,160	115
ST	8	2,702	115
VB	8	3,795	115

Table A.2: Train, validation and test set distribution per domain

Domain	Train Samples	Validation Samples	Test Samples UA
EM	318	112	48
FN	223	100	36
II	152	61	24
LF	277	116	44
LP	51	19	8
ME	582	246	92
MS	165	87	28
MX	496	201	80
PS	376	169	44
SE	376	163	60
ST	188	95	32
VB	274	122	44

Table A.3: Detailed results domain specific learning

Domain	Split	Model Accuracy			Model Weighted F1 Score		
		Validation	Test UA	Test UQ	Validation	Test UA	Test UQ
EM	1	0.770833	0.645833	0.412500	0.778034	0.647980	0.410857
	2	0.678322	0.645833	0.462500	0.654949	0.642350	0.411416
	3	0.706294	0.750000	0.462500	0.701270	0.746205	0.403274
FN	1	0.675926	0.611111	0.300000	0.667502	0.624751	0.215482
	2	0.694444	0.611111	0.275000	0.683495	0.614929	0.180393
	3	0.710280	0.611111	0.425000	0.702417	0.620077	0.409230
II	1	0.464789	0.625000	0.526316	0.450131	0.610979	0.513822
	2	0.577465	0.625000	0.552632	0.567868	0.623684	0.543473
	3	0.619718	0.625000	0.552632	0.621368	0.623684	0.543473
LF	1	0.648855	0.568182	0.300000	0.645209	0.547847	0.374841
	2	0.633588	0.659091	0.425000	0.628214	0.645426	0.513095
	3	0.625954	0.545455	0.425000	0.625092	0.519805	0.507221
LP	1	0.708333	0.750000	0.400000	0.693223	0.722222	0.265566
	2	0.608696	0.750000	0.425000	0.630952	0.763889	0.253509
	3	0.826087	0.875000	0.425000	0.801609	0.888889	0.280423
ME	1	0.793478	0.760870	0.450000	0.789064	0.762836	0.410677
	2	0.731884	0.760870	0.51100	0.729176	0.710420	0.423667
	3	0.746377	0.760870	0.537500	0.737528	0.750420	0.483667
MS	1	0.773810	0.642857	0.425000	0.757454	0.612364	0.471562
	2	0.773810	0.714286	0.400000	0.776347	0.700819	0.389586
	3	0.761905	0.750000	0.500000	0.757559	0.705866	0.402642
MX	1	0.660944	0.575000	0.825000	0.659478	0.575621	0.826587
	2	0.698276	0.600000	0.675000	0.696206	0.594519	0.708988
	3	0.685345	0.650000	0.925000	0.686395	0.645156	0.926111
PS	1	0.851648	0.863636	0.809302	0.849266	0.881667	0.785039
	2	0.890110	0.772727	0.567442	0.884490	0.803776	0.570286
	3	0.911602	0.795455	0.711628	0.911096	0.818023	0.694256
SE	1	0.622222	0.550000	0.325000	0.623976	0.576012	0.315713
	2	0.672222	0.600000	0.425000	0.664733	0.622553	0.426957
	3	0.642458	0.600000	0.400000	0.641860	0.638462	0.373854
ST	1	0.726316	0.562500	-	0.729213	0.592070	-
	2	0.734043	0.593750	-	0.726227	0.622455	-
	3	0.787234	0.687500	-	0.783241	0.708287	-
VB	1	0.659091	0.545455	0.550000	0.646694	0.544108	0.575214
	2	0.674242	0.568182	0.550000	0.679807	0.572808	0.540686
	3	0.666667	0.522727	0.575000	0.645459	0.523088	0.526096

Table A.4: Cheating categories and description

Nr.	Cheating Category	Definition/Description
1	Keywords, meaningless answer	The answer does not make sense and in some cases it is just a concatenation of random- and keywords
2	Keywords, general, unrelated answer	The answer contains identified keywords and makes sense but is not related to the actual question
4	General answer, meaningless argument	The answer is too general and therefore applicable to many things. This makes the argument int he answer meaningless for this context and not related to the actual question
5	Derive solution from question	The answer is derived from the question by using similar words or paraphrasing the question combined with a too general complement
6	Keywords, general, partially true argument but no explanation	The answer contains identified keywords and is too general even though parts of the argument are true the answer does not contain a valid explanation
7	General answer, applicable to many things	The answer is too general and is applicable to many other questions as well
8	Confusing answer, mixing knowledge	The answer is confusion because it contains mixed knowledge where the student only wants to share the related things he knows and is mixing concepts
9	General answer, partially true argument but no explanation	The answer is too general and even though parts of the argument are true the answer does not contain a valid explanation
11	Ambiguous answer	The answer is too general and ambiguous and is only slightly relevant when interpreted in a certain way
12	Correct definition of concept but no valid reason	The answer contains a definition of the concept but does not provide a valid reason
13	Correct answer to opposite question	The answer contains a correct answer to the opposite question